ICIC Express Letters
Volume **14**, Number **9**, September **2020**

ICIC International ©2020 ISSN 1881-803X
pp. 845–854

A NOVEL RESERVATION MODEL BASED ON INTELLIGENT REQUIREMENT ELICITATION TECHNIQUE USING SVM CLASSIFIER

Akram Abdel Qader and Khalil Awad

Department of Software Engineering
Al-Zaytoonah University of Jordan
P.O.Box 130, Airport Street, Amman 11733, Jordan
{ akrama; khalil.awad }@zuj.edu.jo

Received December 2019; accepted March 2020

Abstract. *Most of reservation systems are working on traditional methods and they are user dependent. The requirements engineering process (REP) in these systems also works based on traditional elicitation methods to build the user interface reservation system. These traditional methods are aimed to ensure that systems meet the needs of stakeholders including users, sponsors, and customers. Most reservation systems are based on traditional requirements that meet the stakeholder's needs as they entered the systems. This makes the novel users face some problems relative to their choices in reservation and mostly they did not select the optimal solution. This research proposed a new reservation model based on intelligent requirements elicitation features; these features are stored in a predefined dataset in order to be used in classification process. The support vector machine (SVM) classifier is used to speed up the classification process of finding the optimal solution based on trained intelligent reservation system. The proposed model shows significant improvements in the reservation system based on intelligent feature requirements. The intelligent reservation system is implemented and tested on AL-Zaytoonah University halls and classrooms for reservations of different lectures and meetings. The proposed model shows significant results with high accuracy.*

Keywords: Software requirement, Intelligent requirement elicitation, Requirements classification, Requirements analysis

1. **Introduction.** The online reservation systems have experienced tremendous growth in recent years. The rapid development of communication, information technology and the widespread use of the Internet increase the use of online systems. Such of these systems are the process of accessing large amounts of global data from potential customers on points of interest, travel plans, and destinations [1]. Also, the e-tourism thrives in both the social and economic sector [2]. Hotel recommendation systems in [3] is another important example of smart reservation systems. However, in order to assist customers, many research studies have presented to develop software solutions to provide customers with information that is useful to get best reservation choices [4-7]. Developing efficient software depends on a good recommender system (RS) or filtering approach as well as real-time data communication process [8,9]. In addition, a good recommendation system depends on a good requirements elicitation and analysis. Besides that, the good requirements analysis based on expert users becomes the most important issue to build a good RS. RS systems have many important roles for improving the customers' level of satisfaction [10]. Recently, RS systems assist customers in finding many reservation services such as hotels, tours, tickets, and restaurants by aggregating and analyzing the demographic data of the customers' needs [11]. Furthermore, RS helps improve the customer's loyalty and satisfaction by recommending the acceptable option which is more related to

DOI: 10.24507/icicel.14.09.845

the request of customer [12]. RS is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item. Generally, recommendation systems are classified into four types, which are content-based filtering (CB), collaborative filtering (CF), knowledge-based filtering (KBF), and hybrid filtering, which is a combination of other types [13,14]. Sound Library RS [16], and classroom reservation and management [15] are some of recent applications in reservation systems.

The main contribution of this paper is to present an intelligent reservation model that finds the best solution for the entered requirements features. In order to do that, the following contributions are presented. First, we adapt the elicitation techniques by adding a relationship between requirements features. These requirements features and their relationship are stored in a requirements features table; this table will be used in intelligent reservation model. Second, we present an intelligent reservation model based on an intelligent classification model using support vector machine (SVM) classifier to find the best solution for the user requirements.

The remainders of the paper are organized as follows. Section 2 is related literature review. Section 3 is proposed methodology description and Section 4 shows the experimental results. Section 5 is discussion. Finally, we give conclusions and future works in Section 6.

2. **Literature Review.** Requirements engineering (RE) aims to ensure that systems meet the needs of their stakeholders including users, sponsors, and customers. Often considered as one of the earliest activities in software engineering, it has developed into a set of activities that touch almost every step of the software development process. So many research studies were presented in this field.

In [17] they presented an intelligent simulation conversation Chatbot which is a conversation agent based on a computer program. Chatbot can deal with user inputs in text or voice formats that can be an important role in human-machine interaction. They used three modules: user interface, interpreter, and a knowledge base.

Khan et al. [18] presented a crowd intelligence requirements engineering based on a general research map and suggested the possible future roles. Crowd intelligence requirements help understand the research achievements, the areas of concentration, and how requirements related activities were crowded.

Other researches were proposed by Groen [19], Levy et al. [20] and Srivastava and Sharma [21], who used a macro user communities to elicit requirements using different media channels such as LinkedIn, users forums and research workshops. In recent studies [22] and [23] they developed "Biyubi" and "Requengin" games to complement traditional teaching in order to facilitate the learning of requirements elicitation activities on an undergraduate Software Engineering course. In addition a serious game called "Requengin" has been developed to provide undergraduate students with an interactive learning environment to facilitate the introduction of ISO/IEC/IEEE 29148:2011. These games were developed to explore, analyze, and discover the wishes and needs of the correct stakeholders. In [24] they proposed a personal recommendation system to exploit the user's similarity and familiarity rates. These rates were calculated based on the data that was collected from Facebook using the crawler engine. The recommendation results were based on an acceptance threshold value. Authors [25] proposed content-based recommendation approach based on a learning algorithm using Naive Bayesian classifier using different domains of requirements keywords and simple user demographics. The Bayesian classifier was used based on learned classes as a recommended one. In [26] an intelligent security requirements features model was presented to elicit the security requirement from a text scenario, and the model was based on support vector machine (SVM) classifier to select the most suitable class feature that fit the input requirement scenario. Other research done by Portugal et al. [29] proposed a semi-automated method for finding the possible non-functional requirements in a text based on keywords using asking system stakeholders about a list of qualities. The used catalogs of the non-functional keywords base on the supporting knowledge base. The results performed better in structured text samples to achieve 72% in F1-measure. In [30] they proposed classification and identification method to find non-functional requirements in structured text. They used a set of identification keywords in different texts in identification process, and they achieved results with an average of 57% for F-measure.

Most of the previous studies present manual or semi-automated methods; also they present a recommendation system for specific use. Our proposed method presents a dynamic method for generating system requirement features database based on extracting the features of any system from expert users in the training phase. These features are used to classify the optimal requirements of the system based on SVM classifier.

3. **The Proposed Methodology.** Functional software requirements describe what the system can do and how it should work at the system level. Descriptive software system requirements are the documented results of the intensive software analysis phase as they are collected through various techniques such as interviews, surveys, questionnaire, and sampling methods. Functional specification or software functional requirement focuses defining the system component by describing the functions software must perform such as: inputs, behaviors, and outputs. It can be data manipulation, business process, user interaction, calculation, or any other specific functionality which defines what function a system is likely to perform.

In this paper, we propose a new method to generate a database containing functional system features and functional goals of the input requirements. This database will be the dataset of rich story scenarios of functional requirements or system components. This dataset will be used in the requirement analysis phase to classify and evaluate the requirement statements from user scenarios or questionnaire. The proposed methodology and algorithms are two phases: training phase and testing phase.

The training phase including the following steps:

1) Extract the keywords and features from the requirement statement that are elicited from expert users and stakeholders.
2) Build the table of systems: table of requirement statements and table of keywords and features.
3) Generate the training dataset containing the classes and subclasses of the system requirements keywords and features based on expert users, stakeholders and developers.

The testing phase including the following steps:

1) Input the requirement statement from the user.
2) Extract the keywords and features from the requirement statement.
3) Classify the keywords and features into the most optimal classes in the trained dataset.
4) Use SVM classifier to find the correct classes and subclasses.

The methodology and algorithms of the proposed model are shown in Figure 1.

3.1. **Generating the system requirements table and dataset.** In this stage, we propose an algorithm to create and build the system requirement statements keyword and features to generate the dataset. This dataset can be used in evaluating phase to find the most suitable requirements system from the input user requirement statements. In this section, we generate two tables: table of system names and description (SNDT) as in Table 1, and table of requirement statements, keywords and features (RSKFT) as in Table 2. The system requirements keywords and features are extracted based on expert users and developers to generate the system keywords and features dataset. This table includes the system name, requirements keywords and features in the system.

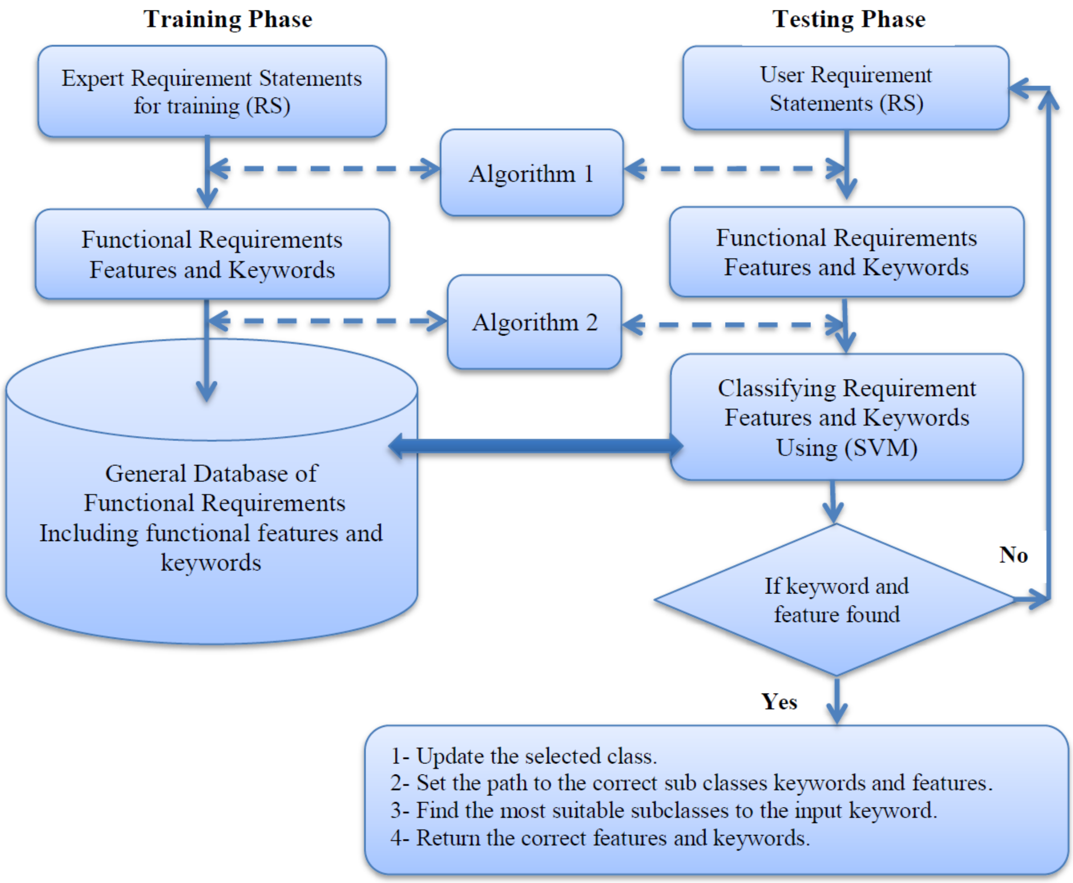

FIGURE 1. The methodology and algorithms of the proposed model

In this algorithm we generate the dataset based on SNDT and RSKFT. This dataset will be trained to be used in the proposed intelligent model to find and evaluate the optimal solution with the most suitable class to the input requirement statements in early stage using the proposed SVM classification process. The algorithm steps are illustrated as the following:

- Create the system name and description table (SNDT) that contains the system number, the system name and the system description as shown in Table 1. This table can dynamically contain any number of systems with its attribute columns.

TABLE 1. System name and description (SNDT)

System number (SN)	System name (SNAM)	System description (SD)

- Create the requirement statements, keywords and features table (RSKFT); this table includes the system name (SNAM), the requirement statements (RS), the requirement keywords (RK) and the requirement features (RF) as shown in Table 2. Each system name has many requirement statements each with relative functional keywords and set of features.

TABLE 2. Requirement statements, keywords and features (RSKFT)

System name (SNAM)	Requirement statements (RS)	Requirements keywords (RK)	Requirements features (RF)
		$Rk_1, Rk_2, Rk_3, \ldots Rk_n$	$Rf_1, Rf_2, Rf_3, \ldots Rf_n$

We considered that these two tables are created by expert users and developer and the information inserted to these tables is tested and trained by expert software engineering developers and users. This dataset table will be used as the base dataset in evaluation process as an early step to find the most suitable requirements to the next requirement statement in the system.

3.2. **Converting the user requirement statements to a set of keywords and features.** In this algorithm we convert the user requirement statements into set of requirement keywords and features. These statements are scanned to extract the functional keywords based on the proposed dataset in Section 3.1. In this step we use a support vector machine (SVM) classifier to detect the functional keyword existence in the saved requirement keywords in trained dataset. The proposed algorithms are illustrated as below.

Algorithm 1: From user requirement statement into set of keywords and features

---

1- Open SNDT and RSKFT dataset tables.
2- Insert the system name and system description.
3- Write the system name and system description into SNDT table.
4- Read the user requirement statement (RS).
  ◦ Correct the user requirement statement for detection errors.
    - Split the user requirement statement into sentences based on (.) dot punctuation mark and remove stop characters by using the general architecture for text engineering (GATE) [27].
    - Extract the words in each RS based on tokenization, sentence splitting and stemming, as shown in classifier design in Figure 2.
    - Normalize the table of keywords to remove the redundancy.
5- For each RS, search for each word in the text using the detection method proposed in Algorithm 2 by comparing the extracted word and features with the trained functional keywords dataset.
  - If the word exists (features found)
    {a. Update the selected class.
    b. Set the path to the correct sub classes keywords and features.
    c. Find the most suitable subclasses to the input keyword.
    d. Return the correct features and keywords as an output result.}
  Else
  {Go to next word.}
6- Repeat steps 4 and 5 until all user requirement statement finished.

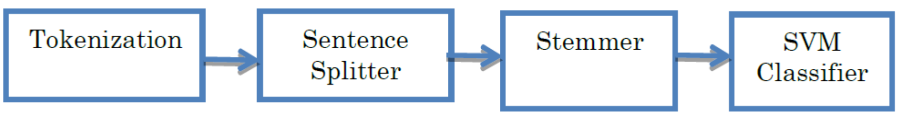

FIGURE 2. The classifier design

In this phase we use an intelligent method to detect and find the most suitable path to the input requirement statement keywords and features based on the trained dataset. This method uses a detection technique based on SVM classifier as shown in Algorithm 2.

Figure 3 illustrates classifying and detecting the correct keywords model using SVM classifier.

Algorithm 2: Classifying and detecting the optimal functional features and keywords using SVM classifier

1- Build machine learning classifier using the trained dataset based on SVM classifier.
2- Convert the input sentences keywords into set of structured features based on the bag of words approach [28]. This approach breaks apart the words in the text keywords into individual word count statistics.
3- Use keywords features to classify the input sentences keywords into the main class and the set of sub classes using SVM classifier.
4- Return the main class and the set of subclass which represent the optimal and the most suitable words to the entered requirement statements.
5- Repeat steps 2 to 4 until RS finished.

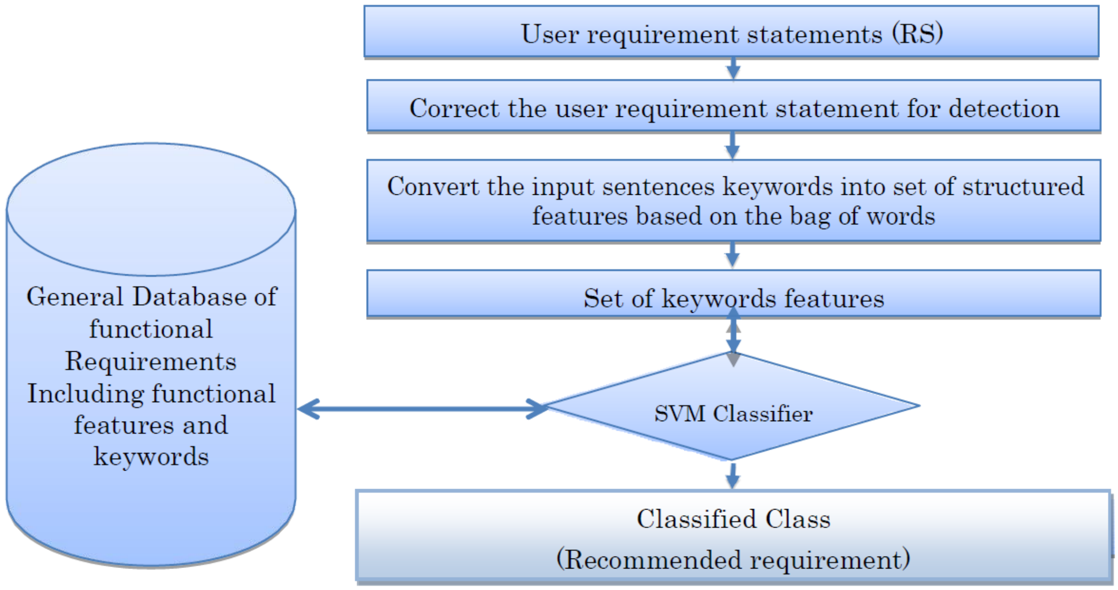

FIGURE 3. Classifying and detecting the correct keywords model using SVM classifier

4. **Experimental Results.** To evaluate the proposed methodology and algorithms we apply a set of experiments based on our generated database. We use Al-Zaytoonah University classrooms reservation requirement statements to collect the user requirements. These statements are input to the proposed intelligent requirement elicitation technique based on SVM classifier. We use a sample of 55% from the available classrooms reservation requirement statements with different class categories in the training phase based on a set of expert users and developers, and we use the remaining 45% of the classrooms reservation requirement statements in the testing phase. This dataset is used to train the proposed model as in Algorithm 1 and Algorithm 2. In our case study Table 3 shows the system number, the system name and the system description, while Table 4 shows the extracted features and keywords for each requirement statement entered from the user.

TABLE 3. The tested user requirements based on the set of RS of classroom reservation

System number (SN)	System name (SNAM)	System description (SD)
1	Classroom reservation	Classroom reservation requirement statements with different classroom categories and needs each room is suitable for specific requirement needs

Table 4. The tested classes for each user requirement based on the set of RS

System name (SNAM)	Requirement statements (RS)	Requirement keywords (RK)	Requirement features (RF)
Classroom reservation user no 1	S_1: Classrooms that meet 3-hours a week are to be scheduled in 50-minute time slots on Sunday, Tuesday and Thursday. S_2: Classrooms that meet 3-hours a week are to be scheduled in 50-minute time slots should have a 10-minute time slots after the end of previous slot on Sunday, Tuesday and Thursday. S_3: Classrooms that meet 3-hours a week are to be scheduled as 50-minute and 10-minute, e.g., 8:00-9:00am, 9:00-10:00am and 10:00-11:00am, 11:00-12:00am, 12:00-13:00am, 13:00-14:00am, 14:00-15:00am, 15:00-16:00am, on Sunday, Tuesday and Thursday. S_4: Classroom should contain 1-desk, 20-seats, 20"chalkboard, and 16"portable-whiteboard. S_5: Classroom should be located in IT College and at the first-floor. S_6: Classroom should be located in IT College with good-lighting. S_7: Classroom can be booking for a meeting or a conference in free date-time. S_8: Classroom should contain 1-desk, 20-seats, 20"chalkboard, and 16"portable-whiteboard and 21-PC and 1-printer. S_9: Classroom should contain 21-PC with Oracle10g-Database and Oracle10g-developer. S_{10}: Classroom should contain 21-PC with Internet-connection.	Rk_1: Classroom, Rk_2: Meet, Rk_3: 3-hours, Rk_4: Week, Rk_5: 50-minute, Rk_6: Time, Rk_7: Slots, Rk_8: Sunday, Rk_9: Tuesday, Rk_{10}: Thursday, Rk_{11}: 10-minute, Rk_{12}: After, Rk_{13}: End, Rk_{14}: Previous, Rk_{15}: Scheduled, Rk_{16}: 8:00-9:00am, Rk_{17}: 9:00-10:00am, Rk_{18}: 10:00-11:00am, Rk_{19}: 11:00-12:00am, Rk_{20}: 12:00-13:00am, Rk_{21}: 13:00-14:00am, Rk_{22}: 14:00-15:00am, Rk_{23}: 15:00-16:00am, Rk_{24}: Should, Rk_{25}: 1-desk, Rk_{26}: 20-seats, Rk_{27}: 20"chalkboard, Rk_{28}: 16"portable-whiteboard, Rk_{29}: Locate, Rk_{30}: IT, Rk_{31}: College, Rk_{32}: First-floor, Rk_{33}: Good-lighting, Rk_{34}: Booking, Rk_{35}: Meeting, Rk_{36}: Conference, Rk_{37}: Free, Rk_{38}: Date-time, Rk_{39}: 21-PC, Rk_{40}: Oracle10g-Database, Rk_{41}: Oracle10g-developer, Rk_{42}: Oracle10g-developer, Rk_{43}: Internet-connection	Rf_1: Classroom, Rf_2: Meet, Rf_3: 3-hours, Rf_4: Week, Rk_5: 50-minute, Rf_6: Time, Rk_7: Slots, Rf_8: Sunday, Rf_9: Tuesday, Rf_{10}: Thursday, Rf_{11}: 10-minute, Rf_{12}: After, Rf_{13}: End, Rf_{14}: Previous, Rf_{15}: Scheduled, Rf_{16}: 8:00-9:00am, Rf_{17}: 9:00-10:00am, Rf_{18}: 10:00-11:00am, Rf_{19}: 11:00-12:00am, Rf_{20}: 12:00-13:00am, Rf_{21}: 13:00-14:00am, Rf_{22}: 14:00-15:00am, Rf_{23}: 15:00-16:00am, Rf_{24}: Should, Rf_{25}: 1-desk, Rf_{26}: 20-seats Rf_{27}: 20"chalkboard, Rf_{28}: 16"portable-whiteboard, Rf_{29}: Locate, Rf_{30}: IT, Rf_{31}: College, Rf_{32}: First-floor, Rk_{33}: Good-lighting, Rf_{34}: Booking, Rf_{35}: Meeting, Rf_{36}: Conference, Rf_{37}: Free, Rf_{38}: Date-time, Rf_{39}: 21-PC, Rf_{40}: Oracle10g-Database, Rf_{41}: Oracle10g-developer, Rf_{42}: Oracle10g-developer, Rf_{43}: Internet-connection

where $S_1, S_2, \ldots, S_n$ are the recruitment statements, and $Rk_1, Rk_2, \ldots, Rk_n$ are the requirement keywords.

These features and keywords are used in classification phase based on SVM classifier and the trained dataset.

The results in Table 5 show the tested classes from class 1 (C1) to class 10 (C10) where each requirements statement feature and keyword is classified using SVM classifier to detect the most suitable class in the training dataset. The result class will be an intelligent solution for the user needs in the classroom reservation system.

The SVM classifier is evaluated based on the tested requirements using 4 user requirement statements from 24 different labs and classrooms. Reservations systems classifier is built based on the 10 different classes. The results are evaluated using the metrics of Precision, Recall and F-Measure [22], defined as follows:

$$Precision = \frac{TP}{TP + FP}, \quad Recall = \frac{TP}{TP + FN}, \quad F\text{-}Measure = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

where TP (True Positive) is the number of correctly classified requirements, FP (False Positive) the number of requirements incorrectly classified, and FN (False Negative) the number of requirements incorrectly not classified.

Table 5. The results of the tested requirement statements keywords classes of classroom reservation system

User requirement	**Number of keywords**	**Number of classified features**	**Requirement statements of feature classes ratio %**									
			C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**	**C10**
User Requirement 1	73	71	2%	3%	0	0	95%	0	0	0	0	0
User Requirement 2	87	85	0	0	3%	8%	0	0	77%	12%	0	0
User Requirement 3	118	111	0	5%	3%	0	2%	50%	0	40%	0	0
User Requirement 4	122	114	10%	0	0	5%	0	40%	0	5%	40%	0
Total	**400**	**381**	**12%**	**8%**	**6%**	**13%**	**97%**	**90%**	**77%**	**57%**	**40%**	**0**

The overall system results based on SVM classifier show significant accuracy of 95.25%, where 381 features were classified and detected from 400 entered features.

5. **Discussion.** This proposed model shows a significant improvement in identifying optimal keywords and features based on the proposed intelligent requirements elicitation features. The proposed model is implemented and tested using a set of different user requirements of classroom reservation system. The proposed model is fully automated method that can be used in eliciting requirements based on intelligent features classification process based on SVM classifier. Our results are compared with a semi-automated process based on structured text called NFRFinder as in [17] it works well over structured text and not tested over unstructured text, the accuracy of F1-measure was 72%. Besides that, our proposed model shows a high performance and a high accuracy rate in F1-measure of 95.25%.

Moreover, our proposed model can classify the requirements into 10 classes based on our generated database. Table 6 illustrates the comparison between our proposed method and the methods proposed in [17] and [30].

Table 6. Comparison between our proposed model and the methods proposed in [17,30]

	Precision	**Recall**	**F-Measure**
The proposed model	97.60%	94.53%	95.25%
Parmar et al. [17]	73.00%	61.00%	72.00%
Cleland-Huang et al. [30]	56.70%	78.9%	57.00%

The proposed method is tested using 10 classes based on our training dataset. In addition, our model can be expandable by adding a new training dataset class for new requirements. Furthermore, our model can be used in different application systems by adding its training dataset to be compatible tool for any user scenario system in the future.

6. **Conclusions and Future Works.** The proposed method will help the developers and system analysts classify the requirements into intelligent features and keywords. In addition, this model will classify the requirements into different classes, and these classes are trained to build a set of optimal intelligent requirements database. This database is

used in testing the new requirements and classifying them based on the trained dataset classes of intelligent requirements keywords and features. The proposed model is tested using several user requirements in classroom reservation system. The results show a high accuracy compared with manual methods and semi-automated methods. In addition, the using of linear SVM classifier improves the accuracy and the performance of the process. The result shows high accuracy rates of 95.25%. This method is a new and novel intelligent approach to identifying and classifying the requirements into set of classes based on intelligent keywords and features. This research opens different future works such as using a different classifier in testing, and other research is to improve the system requirement features to extract the requirement features from direct user speaking. Also many of recommendation systems can be tested using the proposed method.

REFERENCES

[1] L. Sebastia, I. Garcia, E. Onaindia and C. G. Alvarez, e-Tourism: A tourist recommendation and planning application, *Int. J. Artif. Intell. Tools*, vol.18, pp.717-738, 2009.
[2] A. Spencer, *ICT and Caribbean Tourism, in Travel and Tourism in the Caribbean*, Springer, Berlin/Heidelberg, Germany, 2019.
[3] B. Kaya, Hotel recommendation system by bipartite networks and link prediction, *Journal of Information Science*, 2019.
[4] S. Hlee, H. Lee and C. Koo, Hospitality and tourism online review research: A systematic analysis and heuristic-systematic model, *Sustainability*, vol.10, no.4, 2018.
[5] J. Li, Q. Wu, X. Li and D. Zhu, Context-based personalized moblie tourist guide, *Proc. of the IEEE International Conference on Intelligent Computing and Intelligent Systems*, Shanghai, China, 2009.
[6] P. Vansteenwegen, W. Souffriau, G. V. Berghe and D. V. Oudheusden, The city trip planner: An expert system for tourists, *Expert Syst. Appl.*, vol.38, pp.6540-6546, 2011.
[7] D.-Y. Yeh and C.-H. Cheng, Recommendation system for popular tourist attractions in Taiwan using Delphi panel and repertory grid techniques, *Tour. Manag.*, vol.46, pp.164-176, 2015.
[8] L. Ferreira, G. D. Putnik, M. M. Cruz-Cunha and Z. Putnik, Towards effective tourism dynamic packages, *Inf. Resour. Manag. J.*, vol.25, pp.1-21, 2012.
[9] P. Palos-Sanchez, J. R. Saura, A. Reyes-Menendez and I. V. Esquivel, Users acceptance of location-based marketing apps in tourism sector: An exploratory analysis, *J. Spat. Organ. Dyn.*, vol.6, pp.258-270, 2018.
[10] D. H. Park, H. K. Kim, I. Y. Choi and J. K. Kim, Literature review and classification of recommender systems research, *Expert Syst. Appl.*, vol.39, pp.10059-10072, 2012.
[11] J. K. Tarus, Z. Niu and G. Mustafa, Knowledge-based recommendation: A review of ontology-based recommender systems for e-learning, *Artif. Intell. Rev.*, vol.50, pp.21-48, 2018.
[12] L. Etaati and D. Sundaram, Adaptive tourist recommendation system: Conceptual frameworks and implementations, *Vietnam J. Comput. Sci.*, vol.2, pp.95-107, 2015.
[13] J. Borras, A. Moreno and A. Valls, Intelligent tourism recommender systems: A survey, *Expert Syst. Appl.*, vol.41, pp.7370-7389, 2014.
[14] M. A. Mahmood, N. El-Bendary, J. Platoš, A. E. Hassanien and H. A. Hefny, *An Intelligent Multi-Agent Recommender System, in Innovations in Bio-Inspired Computing and Applications*, Springer, Berlin/Heidelberg, Germany, 2014.
[15] R. J. Slapp, III, *Virtual Classroom Management Delivery System and Method*, USPTO Patent Application 20150089376.
[16] J. Smith, D. Weeks, M. Jacob, J. Freeman and B. Magerko, Towards a hybrid recommendation system for a sound library, *IUI Workshops'19*, Los Angeles, USA, 2019.
[17] S. Parmar, M. Meshram, P. Parmar, M. Patel and P. Desai, Smart hotel using intelligent chatbot: A review, *International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT)*, vol.5, no.2, 2019.
[18] J. A. Khan, L. Liu, L. Wen and R. Ali, Crowd intelligence in requirements engineering: Current status and future directions, in *Requirements Engineering: Foundation for Software Quality. REFSQ 2019. Lecture Notes in Computer Science*, E. Knauss and M. Goedicke (eds.), Cham, Springer, 2019.
[19] E. C. Groen, Crowd out the competition gaining market advantage through crowd-based requirements engineering, *IEEE the 1st CrowdRE*, pp.13-18, 2015.
[20] M. Levy, I. Hadar and D. Teeni, A gradual approach to crowd-base requirements engineering: The case of conference online social networks, *IEEE the 2nd CrowdRE*, pp.26-30, 2017.

[21] P. K. Srivastava and R. Sharma, Crowdsourcing to elicit requirements for MyERP application, *IEEE the 1st CrowdRE*, pp.31-35, 2015.

[22] I. Garcia, C. Pacheco, A. León and J. A. Calvo-Manzano, Experiences of using a game for improving learning in software requirements elicitation, *Computer Applications in Engineering Education*, vol.27, pp.249-265, 2019.

[23] I. Garcia, C. Pacheco, A. León and J. A. Calvo-Manzano, A serious game for teaching the fundamentals of ISO/IEC/IEEE 29148 systems and software engineering – Lifecycle processes – Requirements engineering at undergraduate level, *Computer Standards & Interfaces*, vol.67 pp.1-16, 2019.

[24] R.-C. Chen, Hendry and L. Liu, Enhancing personal recommendation system using familiarity factor on social network, *International Journal of Innovative Computing, Information and Control*, vol.13, no.6, pp.2025-2038, 2017.

[25] M. Pazzani and D. Billsus, Learning and revising user profiles: The identification of interesting web sites, *Machine Learning*, vol.27, no.3, pp.313-331, 1997.

[26] A. A. Qader, A novel intelligent model for classify and evaluating non-functional security requirements form scenarios, *Indonesian Journal of Electrical Engineering and Computer Science*, vol.15, no.3, pp.1578-1585, 2019.

[27] H. Cunningham, D. Maynard and K. Bontcheva, *Text Processing with GATE*, Gateway Press, CA, 2011.

[28] A. Zheng and A. Casari, *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*, O'Relly Media, Inc., 2018.

[29] R. L. Q. Portugal, T. Li, L. Silva, E. Almentero and J. C. S. do Prado Leite, NFRFinder: A knowledge based strategy for mining non-functional requirements, *Proc. of the XXXII Brazilian Symposium on Software Engineering (SBES'18)*, São Carlos, São Paulo, Brasil, 2018.

[30] J. Cleland-Huang, R. Settimi, X. Zou and P. Solc, Automated classification of non-functional requirements, *Requirements Engineering*, vol.12, no.2, pp.103-120, 2007.