IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 7, 2018, accepted November 18, 2018, date of publication November 27, 2018,
date of current version December 27, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2883077

Text Categorization Approach for Secure Design Pattern Selection Using Software Requirement Specification

ISHFAQ ALI[1], MUHAMMAD ASIF[1], MUHAMMAD SHAHBAZ[2], ADNAN KHALID[3], MARIAM REHMAN[4], AND AZIZ GUERGACHI[5]

[1]Department of Computer Science, National Textile University, Faisalabad 37610, Pakistan
[2]Department of Computer Science & Engineering, University of Engineering and Technology, Lahore 54890, Pakistan
[3]Department of Computer Science, Government College University, Lahore 54600, Pakistan
[4]Department of Information Technology, Government College University, Faisalabad 38000, Pakistan
[5]Ted Rogers School of Management, Ryerson University, Toronto, ON M5B 2K3, Canada

Corresponding author: Muhammad Asif (asif@ntu.edu.pk)

This work was supported in part by NSERC and in part by CFI.

ABSTRACT Secure patterns provide a solution for the security requirement of the software. There are large number of secure patterns, and it is quite difficult to choose an appropriate pattern. Moreover, selection of these patterns needs security knowledge; generally, developers are not specialized in the domain of security knowledge. This paper can help in the selection of secure pattern on the basis of tradeoffs of the secure pattern using text categorization. A repository of secure design patterns is used as a data set and a repository of requirements artifacts in the form of software requirements specification (SRS) are used for this paper. A text categorization scheme, which begins with preprocessing, indexing of secure patterns, ends up by querying SRS features for retrieving secure design pattern using document retrieval model. For the evaluation of the proposed model, we have used three different domains' SRS. These three SRS documents represent three different domains, i.e., e-commerce, social media, and desktop utility program. A traditional precision and recall method along with F-measure used for evaluation of information/document retrieval model is used to evaluate the results. F-measure for 17 different design problems shows around 81% accuracy with recall up to 0.69%.

INDEX TERMS Design pattern, security, corpus, text categorization, SVD, SVM.

I. INTRODUCTION

Security is considered as a non-functional requirement during software development life cycle. In modern practices of software development life cycle, security requirements are included in each phase of the software development life cycle [1]

As the advancement of technologies has also increased security concerns. In order to cope with security concerns, developers need to learn security requirement of a software and must have security domain knowledge to prescribe a secure development solution. Security concerns or threats are generally categorized into five different measures: Identification and Authentication of users, Access Control mechanisms and Authorization Rules, Cryptography Intrusion Detection and Logging [2]. In [3] defined security requirements properties into four categories i.e. (i) confidentiality, (ii) Integrity, (iii) availability (iv) accountability. This categorization of security concerns, different countermeasures to be taken in order to meet security requirements. Secure development security concerns must be described in a concrete way in each development phase [4]. In the initial stages of development functional requirements are recorded in textual and pictorial forms. These requirements some of the security concerns are explicitly defined; however, there are many security concerns are implicitly defined within functional requirements which are hidden in the plain text. Ignorance of these security concerns can be risky, may harm, the success of the software moreover it can be difficult to retrofit security in an application after development [5]. Learning this security concerns may help to choose an appropriate secure design pattern efficiently and effectively [6]. Due to recurrence of the developmental design problem, a design pattern provides bundled, reusable, tried and test solution for such design problems [7]. These patterns provide experiences and knowledge

of many designers, different repositories are compiled for different types of design problems [8], [9]. Security design patterns were first time presented in [10]. The intentions of these security patterns are to find recurring security requirements and provide a solution which is understandable and applicable even by least security knowledge developers [4]. Secure Design Pattern provides the generic solution of the recurring design problem, in order to choose an appropriate pattern for a specific design problem. As there is a large number of secure patterns available in form of repositories [11]. It is quite difficult to make a decision of selection due to a large number of Secure Design Patterns [12]. Moreover, selection of these patterns needs security knowledge. Commonly developers are not specialized in the domain of security knowledge [13].

The motivation behind the research is to aid the selection of Secure Design Pattern based on their balanced security and performance tradeoff's in accordance requirement of the system learned from the requirements artifacts (System Requirement Specification) using text categorization.

The research question for this research is to investigate effectiveness of text categorization techniques in patterns selection. Moreover, it also investigate the text categorization techniques in terms of learning nonfunctional requirements from unstructured textual artifacts.

A repository of secure design patterns [8] is used as a dataset and a repository of requirements artifacts in the form of software requirements specification (SRS) [14] are used for the research purpose. A text categorization scheme which begins with preprocessing (tokenization, normalization, stemming and lemmatization) followed by indexing of secure patterns (Vector Space Model VSM) and ends up by querying SRS-Features for retrieving secure design pattern using document retrieval model.

In order to evaluate the above-proposed model, we have used three different kind software requirement specification documents (SRS). These three SRS documents represent three different domains i.e. e-commerce, social media, and desktop utility program. A traditional precision and recall method along with F-measure [15] used for the evaluation of the information/document retrieval model is used to evaluate the results. F-measure shows for 27 different design problems shows around 81% accuracy. The organization of this document as followed by Background and Literature Review, Methodology, Evaluation Methodology, Discussion, and Future work.

II. BACKGROUND AND LITERATURE REVIEW

Recently, in the first decade of the 21st century, the range of security patterns repositories had been compiled [6], [8], [17], [18]. As there is now a large number of secure design patterns and increasing day by day. This made it difficult to pick most appropriate security patterns out of the enormous pool of similar patterns. The goal is to choose an appropriate pattern which fulfills security requirements with least performance and cost In reality, deciding on security patterns remains generally an empirical task i.e. You have to choose most suitable among a pool of relevant pattern with some calculation of similarity or suitability. effect [18]. Selection of security patterns from a pool of security patterns approaches is divided into two different methods, Formal Method, and Ontological method. Different efforts in both methodologies exist in the literature are described as follows [19]. Initially, when there were a few numbers of patterns, methods were developed to provide guidelines to choose appropriate patterns. In order to use these methods, developers must have security knowledge as well as the knowledge of software development. As not many developers are expert in security domain knowledge, these guidelines are not helpful for the developers with the least security knowledge. Formal methods provide guidelines for understandings of security requirements and aid selection using UML diagrams [20]. A framework for secure development in which they provide formal guidelines to select the appropriate Pattern [21]. The work of [22] is based on formalizing security patterns in terms of the Goal-Oriented Requirements Language (GRL) and mapping these models into Prolog. Kim and Khawand [23] developed a technique to select the design pattern on the basis of its problem domain specification. In their report, a set of problem domain for design patterns which can be used to define the context of the problem for the design pattern. They used Role-Based Modeling Language (RBML) as UML based formalization of patterns by describing the Meta Model. Meta-Model comprises class collaboration diagrams which describe the structure and behavioral aspects of the classes which can provide the basis for the detection of the patterns. Their approach overcomes the difficulty of UML based structural similarity and scalability variation. However, one limitation of UML based approach it cannot detect the context of the domain. Structural similarity and scalability in terms of a number of design patterns and its variation are the main constraints in the implementation of UML-based approaches [18].

As a large number of Secure Design Patterns are available which made the selection of suitable pattern difficult and time-consuming task. Moreover, the selection process needs scanning through a large number of patterns from the repositories This problem raise the knowledge-based system solution. An ontological approach to find security patterns requested by software developers. This ontological interface contains a mapping between security requirements from one-side and threat models, security bugs, security errors on another side taking into consideration their contexts of applicability [24]. The first one to develop an interface for ontological selection of Secure Design Pattern which is designed for both a high profile secure developer and developer with little security knowledge [16]. However, their works still need a concrete security requirements document and knowledge of Secure Design Patterns. Weiss and Mouratidis [25] Proposed a solution based on the security trade-off required matching with trade-off provided by the security patterns using Goal Oriented Language (GRL) and Prolog. The limitation of their work is it only concerned trade-off related to security

and neglect the performance, manageability, and cost of the overall software. It also needs a properly documented security requirement [26]. Authors produced a solution process of text classification technique (NLP) with comparatively low accuracy solution which analyzes the requirements documents and associated weights to the requirements and use a classifier to select proper selection process [27]. Nevertheless, this solution still needs a document explicitly dealing with security requirements. Moreover, its decision of selection pattern based on similarity, not concerned with the consequences of the pattern applied. Hasheminejad and Jalili [27] presented a text categorization scheme for patterns selection. In their solution, they proposed a two-step solution in which the first step is to learn a classifier for each pattern and the second step is to measure the similarity between design problem and design pattern. The main shortcoming of the work is learning a classifier for each design pattern, which may not feasible for a large number of design patterns. Another shortcoming they used document frequency (DF) as a feature selection method which is not very efficient as compared to other feature selection methods. Moreover, their work suffers from accuracy issues for large sample size and sparse terms. Hussain *et al.* [13] used a sample of textual data and perform different metrics to rank design patterns. In another work Hussain Shahid et al. have developed an information retrieval model for pattern selection. In their prestigious work, they analyzed the effectiveness of multiple feature selections and similarity metrics performance. The main shortcoming of their work they have used sample design problems rather than real-world problems. Aber Hamidy has used Vector Space Model for indexing and measured similarity using cosine similarity for pattern selections.

For the development of secure design pattern selection using text categorization method, software requirement artifacts such as manuals, interviews, expression of interest and most importantly software requirement specification (SRS). These documents are not publically available, because these describe the design of a system which may reveal business secrets or may be a potential security risk for the companies using the system. However, there are a few publically available SRS files of students projects or some companies made it publically available on the internet. Cleland-Haung *et al.* [28] were the first one to compile 15 Software Requirements Specification (SRS). These documents were collected from DePaul University MS student projects. They made it publically available named as ''PROMISE'' Data Set. In their research, they applied TF-IDF (Term Frequency-Inverse Document Frequency) to extract Non-functional requirements. John Slanks et al. included five additional documents healthcare projects. They also labeled 10965 sentences against security and performance attributes. A compiled a data set of 79 software requirements specifications(SRS). Among these 79 SRS files, 49 are taken from industry and 30 are taken from university projects is now available and used for this research [29].

III. METHODOLOGY

A. A BRIEF INTRODUCTION TO INFORMATION RETRIEVAL SYSTEM

Text Categorization or Text Classification (TC) field has gained a great deal of significance in recent years due to the remarkable amount of electronic documents, which are created by different resources such as business organizations, hospitals, insurance agencies, news agencies, and of course academia etc. In an IDC report, it was predicted that the volume of textual data will grow up to 40 petabytes, which will show 50 times growth since 2010 ....[29]. In general, a traditional text classification framework involves pre-processing, feature extraction, feature selection, and classification steps. Although feature extraction, feature reduction and classification algorithm.

Preprocessing is most important and initial components in many Text Classification algorithms. Uysal and Gunal [30] have examined the impact of preprocessing tasks especially in the area of text classification. The preprocessing step usually consists of tasks such as tokenization, normalization or cleaning, lemmatization and stemming. Short description of all these steps is given below. Tokenization is the process of breaking a document into a sequence of characters into pieces (a single word, a pair of words or sentence) called tokens. Some characters are removed at this stage such as symbols and punctuation marks. This list of tokens then subjected to normalization or cleaning of data. Normalization is usually done on documents to remove some unnecessary words. Most common filtering is stop-words removal. Stop words are the high-frequency words which appear in the text without having any significant information (e.g. prepositions, conjunctions, etc.). On the other hand, high-frequency words in the text said to have little information to distinguish different documents and also words occurring at lower frequency are also possibly of non-significant relevance and should be removed from the documents [31]. Lemmatization is a process based on morphological analysis of the tokens, i.e. grouping together the numerous inflected forms of a word, so they can be categorized as a single item. It can also be defined as lemmatization methods attempt to map verb forms to infinite tense and nouns to a single form [32]. Stemming methods goal is to obtaining stem or root of derived words. Stemming is language dependent, therefore, algorithms for stemming are language dependent. There are many stemming algorithms are available but first stemming algorithm was introduced in [30], and most commonly used stemmer is published in [33]. The documents' representation for analysis we need a mathematical model. Most common model is ''Vector Space Model'' (VSM). In VSM documents are represented as numeric vectors [38], [39].

This structure was originally developed for indexing for information retrieval systems. However, it is commonly used for text classification and text mining techniques. VSM provides an efficient and numerical way to analyze large documents. In VSM each word is represented as a numeric value

representing weights of the word. This weight represents the impact of the word on overall learning. There are different weighting techniques such as Boolean, Term Frequency and Term Document Inverse Document Frequency. However, TF-IDF has commonly used weighting technique. In [33] Similarity Measurement is commonly used for document retrieval and information retrieval systems. This similarity is based on vectors derived by using the Vector Space Model. There are different similarity metrics start from the dot product of two vectors to Jaccard similarity. Different similarity metrics are as follows. Inner Product Cosine Similarity Dice Similarity Jaccard Similarity [32] Inner or dot Product: Inner Product is considered as the base for all other similarity metrics. All other similarity metrics are derived from this metric. In Vector Space Model dot product of two vectors is scalar quantity represents angle between two vectors as described in Equation 1.

$$d\,|A.B| = x_1 * x_2 + y_1 * y_2 \tag{1}$$

Cosine is another measure of finding the similarity between two vectors. Which is done by calculating the cosine of the angle between vectors The inner product is based on the distance between two vectors, the cosine is based on the angle between these vectors as described in Equation 2.

$$\cos\theta = \frac{A.B}{|A|\,.\,|B|} = \frac{x_1 * x_2 + y_1 * y_2}{x_1^2 * x_2^2 + y_1^2 * y_2^2} \tag{2}$$

Cosine similarity is most commonly used for information retrieval and document retrieval systems. However, Jaccard and Dice's similarity also provide statistical grounds of measurement for certain domains [35]. The estimation of the future performance of any Information Retrieval systems in a certain domain is a challenging task. There are many algorithms available in the literature, which will be suitable for Information Retrieval System is depends upon it evaluated performance. There are two common evaluation methods used for evaluation of Information Retrieval Systems 1) Scalar Evaluation Method. 2) Visual Evaluation Methods. Scalar Evaluation Method evaluates overall performance; Visual Evaluation Method evaluates comparative performance of an IRS i.e. IRS is better than other. Precision and Recall Curve and Receiver Operator Character (ROC) curve are the example of visual evaluation methods. Precision and Recall, Area Under Curve (AUC) and F measure are the examples of Scalar Evaluation Methods [36].

B. INTRODUCTION TO CORPUS

1) INTRODUCTION TO SOFTWARE REQUIREMENT SPECIFICATION (SRS) CORPUS

On the assumption modern machine learning techniques can lead towards for powerful tool for requirement engineering, many researchers are trying to compile requirements artifacts for different research domains. At DePaul University a collection of 15 requirement specification documents was compiled. However, this compilation has only students projects [5]. But unfortunately, requirements artifacts are not commonly available, due to business secrets and privacy constraints. Ferrari *et al.* [14] compiled a dataset of 79 software requirements specifications(SRS). Among these 79 SRS files, 49 are taken from industry and 30 are taken from university projects. A brief summary of the genre of the repository about their structure is shown in Figure 1.

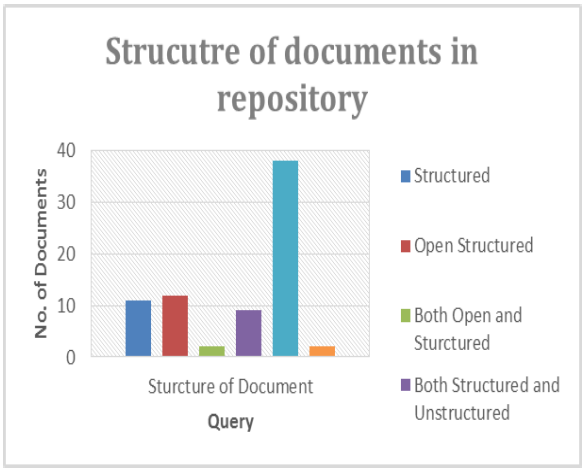


FIGURE 1. Structure of documents in repository.

Files format for these SRS files 62 PDF, doc 13, HTML 3 and one RTF file format. Unfortunately, this dataset contains images which may carry significant data which is not included in proposed methodologies..

2) CORPUS OF THE SECURE DESIGN PATTERN

Compilations of security patterns, organizing is another research topic, research since from the inception of the idea of security patterns [2]. Different security patterns repositories are available in the literature [6]. Although there 481 reported security pattern on the only active online security patterns repository at [37].

In their work the actively scan the literature to find security patterns according described by the researcher in literature. However, I have employed Kienzle *et al.* [8] security patterns repository due to a number of reason1) deliberately defined the context of the patterns 2) Defined genre of the pattern. 3) Well defined security patterns tradeoffs. The overall structure of the corpus 26 Structural pattern, 26 procedurals, and a few mini patterns

C. PREPARATION OF DATA SET

1) PREPARATION OF SRS FILES

For the purpose of indexing, authors have used a repository of publically available SRS files; which includes SRS files documented in from 1999 to 2011 years. The initial decision is to choose what should be indexed from large documents and unstructured documents having more than 15000 words. As Table of contents, Credits, Acknowledgment, List of figures, List of tables are excluded during the indexing process. Since the amount of text in these SRS files is large which

TABLE 1. Structure of SRS query.

Heading	Explanation
Sr. No	Serial Number of SRS
Name	SRS name
Domain	Domain as Described in SRS
Purpose/Scope	Purpose and scope of project textual data
F1 F2 . . F10	Functional Modules Textual Data (Maximum Module in the repositories in all SRS is ten but most of Project have less than 10 modules. Null is used for SRS having less than 10 modules.
NFR	Non Functional Requirements (if formally Encoded otherwise Null)

includes a lot of irrelevant text which is excluded from these documents. We employed a selection of SRS for querying a criterion as it must be documented from 2009 to 2011 and have less than 10 modules. SRS contains multiple sections, but we have indexed sections which describe overall and individual module intent related security properties. Sections which describes intents of the systems are Introduction, Purpose/ Scope and detail description of the specification of functionalities. Some of SRS includes a formal description of nonfunctional requirements. The overall structure of resulting query corpus as given in the table1.

2) PREPARATION OF SECURITY PATTERNS REPOSITORY

Secure Design Patterns in the repository have multiple sections and subsections of text, choosing what should be included and what should be excluded is a tedious task. However, in order to achieve maximum unique words following sections of each pattern from the repository as follows

A CSV (Comma Delimited) sheet is prepared which includes 23 structural secure design patterns, 39 procedural secure design pattern and 11 mini patterns with UTF-8 Encoding Scheme. As given in the following table2.

D. DOCUMENT RETRIEVAL MODEL FOR PATTERN SELECTION

Text Preprocessing begins with tokenization of sentences i.e. conversion of long sentences into short tokens, for creating tokens of sentences. Then these sentences are subjected to the Normalization process. Normalization process includes multiple steps such as remove numbers, remove punctuation, remove symbols, strip white space and finally changing the case. Finally, stop words are removed which carries no information whole process is shown in Figure 2.

This Normalized data undergoes stemming process, the process of eliminating affixes (suffixed, prefixes, infixes,

TABLE 2. Secure design pattern indexing.

Heading	Explanation.
Pattern_Id	For Identification of pattern in a repository
Pattern_Name	Name of Secure Design Pattern
Abstract	Abstract of respective Secure Design Pattern
Problem	Representing Design Problem
Solution	Solution of secure Design Patterns.
Tradeoffs	Encoded by the authors. Otherwise Null

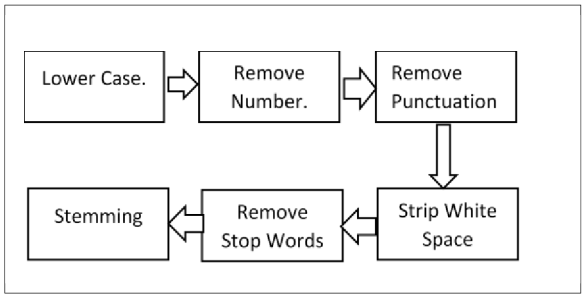


FIGURE 2. Preprocessing of text.

circumfixes) from a word in order to obtain a word stem. This Stemmed data is lemmatized for getting cleaned data. Lemmatization is similar to stemming, but lemmatization is able to capture canonical forms based on a word's lemma and more efficient in terms of cleaning data. For example, stemming the word "better" would fail to return its citation form (another word for lemma); however, lemmatization would result in the following: better →good. VSM is a common indexing method for text categorization, VSM is an algebraic model for the representation of textual documents commonly used for Information Retrieval Systems, Indexing.

For SRS files feature vectors of Purpose/Scope, Functional Requirements, and Non-Functional Requirements are indexed. For Secure Design Patterns Abstract, Problem, and Solution is converted into a single feature vector. However, Tradeoffs is indexed as another feature vector. VSM model represents each secure design pattern as a vector of features. These features are unigrams or single words and bigrams or a pair of consecutive words. Bigram features are significant to differentiate between the different security patterns category. For describing documents, preprocessed data is indexed. In order to index Vector Space Model (VSM) is used. There are three different categories of VSM such as term-document category, word-context category, and pair pattern category. However, the term-document category is a widely used technique for indexing. In this research, we have also

use the term-document category. In VSM term-document category each document is described over-occurrence of a term in a document. Which can be described mathematically. $D = (term)_w$ here "term "is the Term in document d where weights can be calculated using different methods such as Binary, Term Frequency (F), Term Frequency Collection (TFC), Length Term Collection (LTC), Term Frequency-Inverse Document Frequency (TFIDF), and Entropy weighting. However, TFIDF is most common the and effective method for weighting and in this research, it is used. In order to remove features, different methods can be used. These methods are categorized into three categories such as wrappers, filters and embedded methods. However, selection of feature selection method based on computational time and classification accuracy. The wrapper and embedded method are not suitable for text categorization as they interact classifier during flow which may increase the running time for higher dimensional data. Filter based approach is commonly used in text categorization due to its efficiency and low running cost. Filter based techniques include Information Gain, Document Frequency, Chi-Square (CHI), Correlation Coefficient (CC), Mutual Information (MI) etc.

As discussed earlier there are different weighting schemes are available, however, we have used the Term Frequency-Inverse Document as it is the most common method. TFIDF has been used for a similar problem. Hussain *et al.* [13] finds it most efficient weighting method. Description of TFIDF method calculation as described in Figure 3.

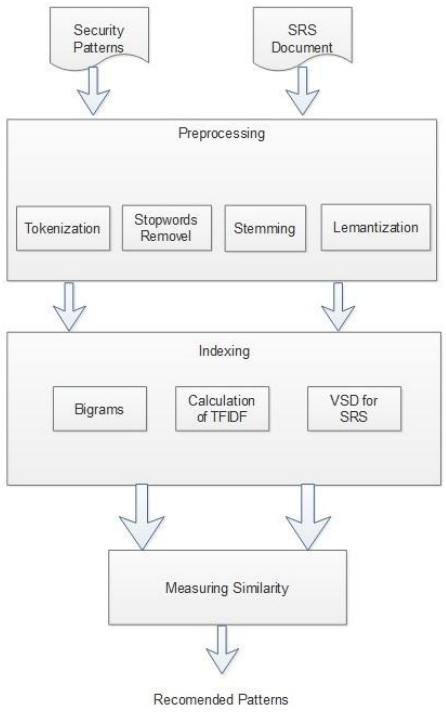


FIGURE 3. Proposed methodology.

Term Frequency represents a number of occurrences of a certain term in a document. Let f(t, d) represents the frequency of an individual term t in document d and TF(t,d) represents the proportion of the count of term T in document d so that it can be represented mathematically as described in Equation 3

$$TF(t_i, d) = \frac{f(t, d)}{\sum_i^n f(t_i, d_i)} \tag{3}$$

Let us suppose a number of documents in the repository are N and C is the count of documents in which term t appears at least once. The mathematical formula for calculation of Inverse Document Frequency as described in Equation 4.

$$IDF\ (T) = \log_{10}\left(\frac{N}{C\ (t_i)}\right) \tag{4}$$

Term document inverse document frequency is calculated with the multiplication of a factor

TF and IDF can be calculated as described in Equation 5.

$$TF_{IDF} = TF\ (t_i) \times IDF\ (t_i) \tag{5}$$

Singular Value Decomposition is a well-known method for text categorization, noise removal, and dimension reduction. The application of Singular Value Decomposition (SVD) in a document-term vector space model has been proposed in [32] Documents and patterns are represented with vectors and SVD is applied for reducing the dimensions of these vectors [36]. Let us take Pattern_Repositry with a matrix of order p∗q, first step to apply SVD is convert into two orthogonal matrices and a diagonal matrix as described in Equation 6.

$$Rep_{p\times q} = U_{p\times p} \times S_{p\times q} \times V^t_{r\times r} \tag{6}$$

Here $Rep_{p\times q}$ represents repository of secure design patterns with The columns of U are orthonormal eigenvectors of AA^T, S is a diagonal matrix containing the square roots of eigenvalues from U or V in descending order and the columns of V are orthonormal eigenvectors of AT A. To remove some noise from the data, dimensionality reduction should be applied. This is done by removing rows from the bottom of matrices U and S and left columns from matrices S and V T. In order to choose a pattern, similarity must be calculated between query i.e. SRS_Query with N features with security patterns repository as described in 4.

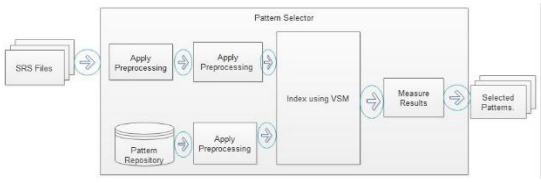


FIGURE 4. Pattern reterival model.

There are many metrics to calculate similarity simple dot product to trigonometric similarity. One of the most important similarity metrics is cosine similarity used for information retrieval system. The cosine similarity between two documents on VSM is a measurement of the cosine of the angle between them. Cosine Similarity is a metric of measurement of angle or orientation and not magnitude. This metric

provides a comparison between documents on a normalized space. The equation to compute Cosine similarity between query q and p Pattern Repository as described in Equation 7

$$cosine\ similarity = \frac{\sum_{i=1}^{n}(p_i \times q_i)}{\sqrt{\sum_{i=1}^{n} p_i^2 \times \sum_{i=1}^{n} q_i^2}} \quad (7)$$

As there are three different types of patterns (Structural, Procedural and Mini Patterns.) in the repository, therefore three different types of quires tuned to predict suitable pattern against Requirement Specification as described in SRS file For structural patterns selection, the overall intent of the Security and Performance must be learned. For learning overall intent, a matrix is created which includes vectors Purpose/ Scope, F1 to F10 vectors and NFR vectors. This process can be described as pseudo code. As procedural patterns are used for an individual activity such as login authentication of users so each functional requirement is from F1 to F10 is passed along with NFRs as individual query and patterns similarity is calculated for each functional requirements. In Querying for Mini patterns a matrix which includes vectors NFRs and related functionality.

IV. EVALUATION METHODOLOGY

The estimation of the future performance of any Information Retrieval systems in a certain domain is a challenging task. There are many algorithms available in the literature, which will be suitable for Information Retrieval System is depends upon it evaluated performance. There are two common evaluation methods used for evaluation of Information Retrieval Systems 1) Scalar Evaluation Method. 2) Visual Evaluation Methods [36]. Scalar Evaluation Method evaluates overall performance; Visual Evaluation Method evaluates comparative performance of an IRS i.e. IRS is better than other. Precision and Recall Curve and Receiver Operator Character (ROC) curve are the example of visual evaluation methods. Precision and Recall, Area Under Curve (AUC) and F measure are the examples of Scalar Evaluation Method [34]. This research employs Precision and Recall Evaluation Method along with F-measure. Detail description evaluation of Precision and Recall method as follows.

A. PRECISION AND RECALL METHOD

Precision and Recall have frequently used measures for the effectiveness of an Information Retrieval System. In terms of pattern selection information system Precision can be termed as the fraction of retrieved items that are relevant to all retrieved items or the probability given that a pattern is retrieved it will be relevant and recall as the fraction of relevant items that are retrieved to relevant items in the repository or the probability given that a pattern is relevant it will be retrieved [38]. These terms are illustrated in Figure 5.

The formula for calculating Precision as described in Equation 8

$$precision = \frac{n(A \cap B)}{n(B)} \quad (8)$$

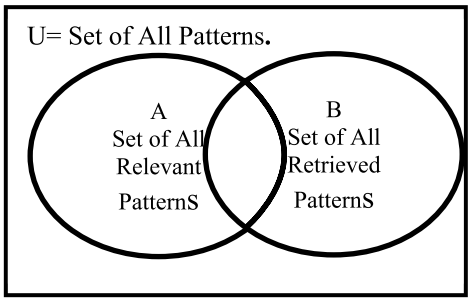


FIGURE 5. Precision and recall method.

and for calculating recall as follows as described in Equation 9.

$$Recall = \frac{n(A \cap B)}{n(A)} \quad (9)$$

F-measure for Precision and Recall is as described in Equation 10

$$F = \frac{1}{\alpha \times \frac{1}{Precesion} + (1-\alpha) \times \frac{1}{Recall}} \quad here\ value\ of\ \alpha[0,1]$$
$$however\ we\ have\ take\ optimal\ value\ of\ 0.5 \quad (10)$$

There is another measure known as fallout which can be used for a situation in which no relevant document retrieved, no relevant document in the dataset [14]. The formula for calculating Fallout is as described in Equation 11.

$$Fallout = \frac{n}{m} \quad here\ n\ is\ the\ number\ of\ non\ relvant\ document\ reterived\ and\ m\ shows\ total\ non\ relevant\ docment\ in\ the\ repository \quad (11)$$

For a single SRS query, these all parameters can be calculated as follows.

Here $R_p$termed as the ranking position of retrieved patterns. Where retrieved patterns section indicates the identity of retrieved patterns such P_12 describe 12^{th}Patterns in the repository. You can observe that the value of Recall does not change for the 4^{th}Ranked pattern.

1) THE ASSUMPTION FOR EVALUATION

If a pattern is applied in the current version, all the related patterns in a repository are considered as relevant patterns. For example, ''Authenticated Session'' is used in the current version there are six related patterns which can be used alternatively are considered as a relevant pattern. If the retrieved pattern is relevant, then F-measure will be calculated otherwise Fallout measure will be calculated. If the current version is using a variant of parent pattern, then parent pattern and all its related pattern in the repository are considered as relevant patterns.

B. EVALUATION OF PROJECTS

1) EVALUATION OF RESULTS FOR SRS_13: PDF SPLIT-MERGE

Design of Problem. This SRS document describes a utility software for pdf files format. It provides multiple services

TABLE 3. Description of Project.

SRS id	01
Name of Project	PDF Split and Merge
Genre of Project	Desktop Utility
Domain	Utility Software
Interfaces/Modules	User Interfaces - GUI Hardware Interfaces Software Interfaces Communications Interfaces
Functional Requirements	1. Split. 2. Merge/Extract 3. Alternate Mix 4. Rotate 5. Visually Reorder 6. Visually Compose 7. Working Environment 8. Log Panel 9. Settings
Encoded Non Functional Requirements	1. Performance Requirements. 2. Safety Requirements. 3. Security Requirements. 4. Software Quality Attributes. 5. Legal Requirement which GPU licenses.

regarding PDF file format. This project is a desktop utility software with the least security requirement. The SRS file in the document was written for the first version of the software. However, its resources are available for its current version and its documentation from [39] Detail Query for its design problem is given below.

2) QUERY FOR STRUCTURAL PATTERNS

As structural pattern selection requires overall intent of the security and performance requirements. Therefore, the design problem for structural design pattern includes a project scope, functional requirements, and non-functional requirements. Query for pattern selection as follows

3) QUERYING FOR FR-1 (DOCUMENT SPLIT)

The first Feature of this software is the ability to split a pdf document into chapters, even individual pages. A vector which describes this functional requirement with 740 words and 346 sentences. The query for procedural pattern selection for the functional requirement as given Query 1.

$$Q\,(SRS_{13} - FR_1) = Vector\,(Scope, FR_1, NFR_1, NFR_2, NFR_3) \quad \text{(Query 1)}$$

4) QUERYING FOR FR-2 (MERGE/EXTRACT)

This feature requirement provides users with ability to merge multiple documents into single documents. It also provides ability to extract certain data or certain pages from a single document. Vector for these requirements have 445 words and 213 sentences. Query for selection of procedural patterns as given in Query 2.

$$Q\,(SRS_{13} - FR_2) = Vector\,(Scope, FR_2, NFR_1, NFR_2, NFR_3) \quad \text{(Query 2)}$$

5) QUERYING FOR FR-3 (ALTERNATE MIX.)

This feature requirement provides users with the ability to mix multiple pdf documents into a single document. It also enables the user to mix pages without any order i.e. mixing pages' page number 23 and page number 45 into page number 51 and 52. Vector for this feature requirement has 380 words and 119 sentences. Query for this feature requirement as given described in Query 3.

$$Q\,(SRS_{13} - FR_3) = Vector\,(Scope, FR_3, NFR_1, NFR_2, NFR_3) \quad \text{(Query 3)}$$

6) QUERYING FOR FR-4 (ROTATE PAGES)

This feature enables users to rotate individual page as well as whole pdf document in single stimulus. However, users cannot rotate individual pages from bulk of page i.e. user can't rotate page number 13 out of hundred pages' document. Vector have 319 words and 89 sentence and query for pattern selection as described in Query 4.

$$Q\,(SRS_{13} - FR_4) = Vector\,(Scope, FR_4, NFR_1, NFR_2, NFR_3) \quad \text{(Query 4)}$$

7) QUERYING FOR FR-5 (VISUAL REORDER)

This feature enables to rotate, reorder and delete specific pages from a pdf document. It provides a visual interface for the above-mentioned task. This feature vector has 556 words and 324 sentences and the query for this feature described in Query 5.

$$Q\,(SRS_{13} - FR_5) = Vector\,(Scope, FR_5, NFR_1, NFR_2, NFR_3) \quad \text{(Query 5)}$$

8) QUERYING FOR FR-6 (VISUALLY COMPOSE)

Design Problem. This system feature allows the user to utilize a GUI interface to rotate, reorder and delete specific pages from a pdf document or multiple documents. It allows all feature such as split, merge, rotate and other features of the software. The Vector for this feature has 410 words and 91 sentences, query for pattern selection as described in Query 6.

$$Q\,(SRS_{13} - FR_6) = Vector\,(Scope, FR_6, NFR_1, NFR_2, NFR_3) \quad \text{(Query 6)}$$

9) QUERYING FOR FR-7 WORKING ENVIRONMENT

This feature allows the users to save and reload its working environment so that he/she can resume discontinued work. This Feature's vector contains 114 words and 13 sentences. Query for pattern selection is as described in Query 7.

$$Q\,(SRS_{13} - FR_7) = Vector\,(Scope, FR_7, NFR_1, NFR_2, NFR_3) \quad \text{(Query 7)}$$

Precision and Recall table for pattern selection as follows. Precision and Recall Plot for Queries as shown in the figure5

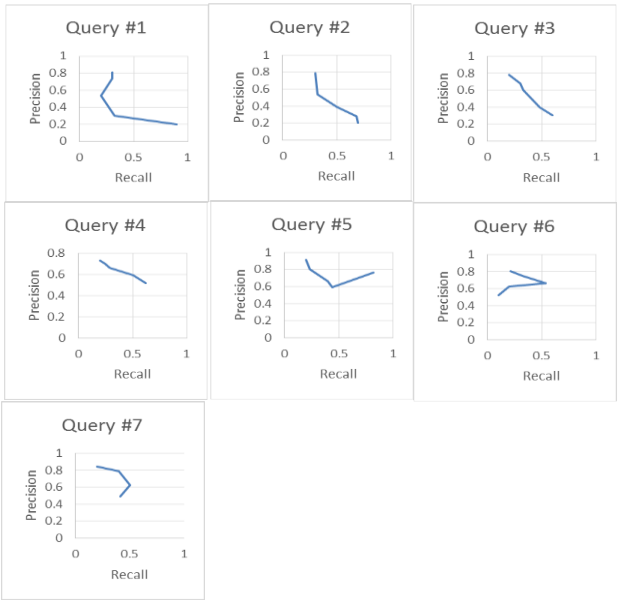


FIGURE 6. Precision and recall plot.

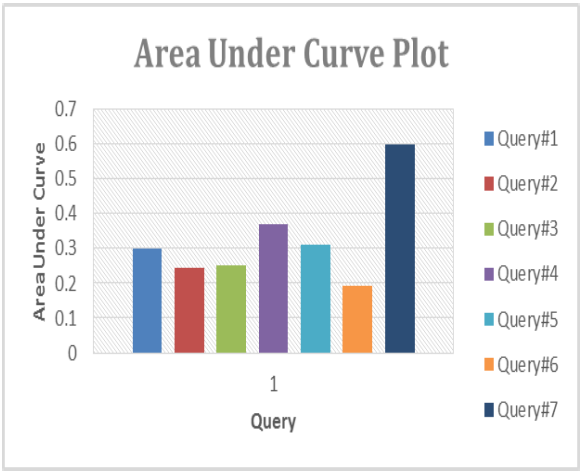


FIGURE 7. Area under curve.

and Precision and Recall Plot is in figure 6 and area under curve is shown in figure7

Discussion. Reason to select this Project for evaluation is to find out how effectively it can produce results with systems having least or no security requirements. It was stated in the SRS for the version-I there is security requirement and no need of any security requirements however if we examine current version i.e. V2.0 in which a structural pattern ''Authenticated Session'' is implemented. The results shown in the form of precision and recall table shows how effectively it can learn implicit security requirements. As there is some pattern ranked which are the variant of ''Authenticated Session''.

C. EVALUATION OF RESULTS FOR SRS_9 OF PROJECT MASHBOOT

See Table 4.

TABLE 4. Project description.

SRS id	09
Name of Project	Mashboot for the enterprise.
Platform of Project	Web Based
Domain	Social Media
Number of Modules and their names	• External Interface- • Email System • Internal Interface for User.
Number of Functional Requirements	• User Accounts • Marketing Campaigns. • External Accounts Services.
Encoded Non Functional Requirements	Yes

D. FR_Id # USER ACCOUNT

This feature includes users account creation, login screen, and interfaces according to their role. There are three different roles of the user's Publisher, Contributor and Approver, and development of three interfaces. In order to query for pattern selection, the query should be broken in accordance with all three users as described in Query 8,9 and 10.

$$Q\,(SRS_9 - U_1) = Vector\,(Scope, U_1, NFR_1, NFR_2, NFR_3) \quad \text{(Query 8)}$$

$$Q\,(SRS_9 - U_2) = Vector\,(Scope, U_2, NFR_1, NFR_2, NFR_3) \quad \text{(Query 9)}$$

$$Q\,(SRS_9 - U_3) = Vector\,(Scope, U_3, NFR_1, NFR_2, NFR_3) \quad \text{(Query 10)}$$

1) QUERY FOR ACCOUNT CREATION

Account Creation is a feature for registering a new user and maintain his/her profile. The design problem is the creation of new users, setting up their role in accordance with their position and marketing products. Vector for Account Creation contains 214 words and 89 sentences and the query for account creation as described in Query 11,12 and 13.

$$Q\,(SRS_9 - AU_1) = Vector(Scope, AU_1, U_1, NFR_1, NFR_2, NFR_3) \quad \text{(Query 11)}$$

$$Q\,(SRS_9 - AU_2) = Vector(Scope, AU_2, U_2, NFR_1, NFR_2, NFR_3) \quad \text{(Query 12)}$$

FIGURE 8. Precision and recall plot.

$$\mathbf{Q\,(SRS_9{-}AU_3)} = \mathbf{Vector(Scope, AU_3, U_3, NFR_1, NFR_2, NFR_3)} \quad \text{(Query 13)}$$

2) QUERY FOR SECURITY PATTERN EXTERNAL USERS

As there are two types of users internal and external and internal users are further categorized into three categories i.e. Contributor, Approver, and Publisher. Security needs for internal and external users vary so that pattern varies. Query for this pattern selection includes both vectors of external and internal users described in Query 14,15 and 16.

$$Q\,(SRS_9 - EU_1) = Vector(Scope, EU_1, U_1, NFR_1, NFR_2, NFR_3) \quad \text{(Query 14)}$$

$$Q\,(SRS_9 - EU_2) = Vector(Scope, EU_2, U_2, NFR_1, NFR_2, NFR_3) \quad \text{(Query 15)}$$

$$Q\,(SRS_9 - EU_3) = Vector(Scope, EU_3, U_3, NFR_1, NFR_2, NFR_3) \quad \text{(Query 16)}$$

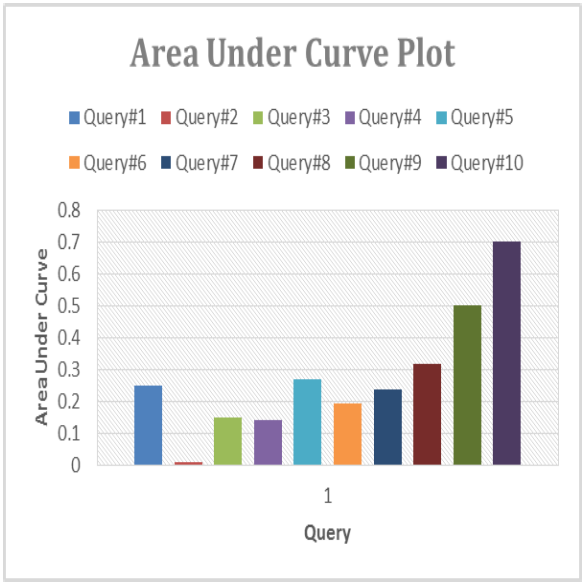


FIGURE 9. Area under curve.

3) QUERY FOR INTERNAL USERS

Internal user's security requirements vector has three different user's requirements.

This vector has 458 words and 114 sentences. The query is

$$Q\,(SRS_9{-}IU_1) = Vector\,(Scope, IU_1, , NFR_1, NFR_2, NFR_3) \quad \text{(Query 17)}$$

Precision and Recall Table is shown in Figure 8 and Area Under Curve is shown Figure 9

V. CONCLUSION AND FUTURE WORK

The proposed text categorization approach based on document retrieval model which is evaluated with respect to Software requirement specification which is publically available. The proposed approach shows the good result with F1 measure up to 81%. The proposed approach provides an easier way to select a suitable security pattern compared to other approaches based on UML, Ontology, and Text categorization approaches. As you need to break your functional requirement into an individual design problem to query from the model. Most of pattern selection approaches are evaluated on the basis on certain design problem with low sparsity terms meanwhile in real-world requirements are encoded with respect to their domain and their own wording which creates a requirement feature with higher ratio sparsity terms. This approach provides a solution which can be applied with high sparsity terms. Another contribution of the proposed approach is the development of an indexed repository of SRS files and Secure design pattern repository. This approach can be used to select design patterns, which require a properly indexed design pattern repository.

In future work, we will consider unsupervised techniques for pattern selection, and improvement in proposed methodology by increasing the feature vector size and comparison of different techniques for reduction of sparsity terms.

VI. ACKNOWLEDGEMENT

The authors would like to thank Dr. M. S. Sarfaraz, Associate Professor, FAST, National University of Computer and Emerging Science, for the useful technical comments.

REFERENCES

[1] A. Souag, C. Salinesi, R. Mazo, and I. Comyn-Wattiau, "A security ontology for security requirements elicitation," in *Engineering Secure Software and Systems* (Lecture Notes in Computer Science: Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 8978. Cham, Switzerland: Springer, 2015.

[2] M. D. Abrams, "Security engineering in an evolutionary acquisition environment," in *Proc. New Secur. Paradigms Workshop*, 1998, pp. 11–20.

[3] M. Schumacher, *Security Engineering With Patterns*, vol. 2754. Berlin, Germany: Springer-Verlag, 2003.

[4] M. Schumacher, E. Fernandez-Buglioni, D. Hybertson, F. Buschmann, and P. Sommerlad, *Security Patterns: Integrating Security and Systems Engineering*. Hoboken, NJ, USA: Wiley, 2006.

[5] M. Riaz, J. King, J. Slankas, and L. Williams, "Hidden in plain sight: Automatically identifying security requirements from natural language artifacts," in *Proc. IEEE 22nd Int. Requirements Eng. Conf. (RE)*, Aug. 2014, pp. 183–192.

[6] I. Duncan and J. de Muijnck-Hughes, "Security pattern evaluation," *Proc. IEEE 8th Int. Symp. Service Oriented Syst. Eng. (SOSE)*, Apr. 2014, pp. 428–429.

[7] S. Ouhbi, A. Idri, J. L. Fernández-Alemán, and A. Toval, "Requirements engineering education: A systematic mapping study," *Requirements Eng.*, vol. 20, no. 2, pp. 119–138, 2015.

[8] D. M. Kienzle, M. C. Elder, D. Tyree, and J. Edwards-Hewitt, "Security patterns repository version 1.0," DARPA, Arlington, VA, USA, Tech. Rep., 2002.

[9] C. Steel, R. Nagappan, and R. Lai, "Core security patterns," Sun Microsyst., Santa Clara, CA, USA, 2006.

[10] J. Yoder and J. Barcalow, "Architectural patterns for enabling application security," *Urbana*, vol. 51, p. 61801, 1998.

[11] E. B. Fernandez and R. Pan, "A pattern language for security models," in *Proc. 8th Conf. Pattern Lang. Programs*, 2001, pp. 1–13.

[12] T. Li, C. Zhang, and M. Ogihara, "A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression," *Bioinformatics*, vol. 20, no. 15, pp. 2429–2437, 2004.

[13] S. Hussain, J. Keung, and A. A. Khan, "Software design patterns classification and selection using text categorization approach," *Appl. Soft Comput. J.*, vol. 58, pp. 225–244, Sep. 2017.

[14] A. Ferrari, G. O. Spagnolo, and S. Gnesi, "Towards a dataset for natural language requirements processing," in *Proc. CEUR Workshop*, vol. 1796, 2017, pp. 1–6.

[15] F. Sebastiani, "Machine learning in automated text categorization," *ACM Comput. Surv.*, vol. 34, no. 1, pp. 1–47, 2002.

[16] M. Hafiz, P. Adamczyk, and R. E. Johnson, "Organizing security patterns," *IEEE Softw.*, vol. 24, no. 4, 2007.

[17] B. Biggio, G. Fumera, and F. Roli, "Security evaluation of pattern classifiers under attack," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 4, pp. 984–996, Apr. 2014.

[18] A. V. Uzunov, K. Falkner, and E. B. Fernandez, "A comprehensive pattern-oriented approach to engineering security methodologies," *Inf. Softw. Technol.*, vol. 57, pp. 217–247, Jan. 2015.

[19] J2EE-SecPatterns. *Security Pattern Catalog*. Accessed: Oct. 23, 2018. [Online]. Available: https://people.cs.kuleuven.be/~koen.yskout/icse15/catalog.pdf

[20] K. Beckers, M. Heisel, and D. Hatebur, *Pattern and Security Requirements*. Springer, 2015.

[21] A. Motii, A. Lanusse, B. Hamid, and J.-M. Bruel, "Model-based real-time evaluation of security patterns: A SCADA system case study," in *Computer Safety, Reliability, and Security* (Lecture Notes in Computer Science: Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 9923. Cham, Switzerland: Springer, 2016, pp. 375–389.

[22] M. Weiss and H. Mouratidis, "Selecting security patterns that fulfill security requirements," in *Proc. Int. Requirements Eng.*, Sep. 2008, pp. 169–172.

[23] D.-K. Kim and C. El Khawand, "An approach to precisely specifying the problem domain of design patterns," *J. Vis. Lang. Comput.*, vol. 18, no. 6, pp. 560–591, 2007.

[24] P. El Khoury, A. Mokhtari, E. Coquery, and M.-S. Hacid, "An ontological interface for software developers to select security patterns," in *Proc. 19th Int. Workshop Database Expert Syst. Appl.*, 2008, pp. 297–301.

[25] M. Weiss and H. Mouratidis, "Selecting security patterns that fulfill security requirements," in *Proc. 16th IEEE Int. Requirements Eng. Conf. (RE)*, vol. 8, Sep. 2008, pp. 169–172.

[26] A. Van Den Berghe, S. Van Baelen, Y. Berbers, W. Joosen, and J. Van Haaren, "Towards an automated pattern selection procedure in software models," in *Proc. Int. Conf. Inductive Logic Program.*, 2012, pp. 68–73.

[27] S. M. H. Hasheminejad and S. Jalili, "Design patterns selection: An automatic two-phase method," *J. Syst. Softw.*, vol. 85, no. 2, pp. 408–424, 2012.

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

[29] A. Ferrari, G. O. Spagnolo, and S. Gnesi, "Towards a dataset for natural language requirements processing," in *Proc. REFSQ Workshops*, 2017, pp. 1–6.

[30] A. K. Uysal and S. Gunal, "The impact of preprocessing on text classification," *Inf. Process. Manag.*, vol. 50, no. 1, pp. 104–112, 2014.

[31] H. Saif, M. Fernández, Y. He, and H. Alani, "On stopwords, filtering and data sparsity for sentiment analysis of Twitter," *Proc. 9th Int. Conf. Lang. Resour. Eval.*, 2014, pp. 810–817.

[32] S. Vijayarani, J. Ilamathi, and M. Nithya, "Preprocessing techniques for text mining—An overview," *Int. J. Comput. Sci. Commun. Netw.*, vol. 5, no. 1, pp. 7–16, 2015.

[33] J. J. Webster and C. Kit, "Tokenization as the initial phase in NLP," in *Proc. 14th Conf. Comput. Linguistics*, vol. 4, 1992, pp. 1106–1110.

[34] C. De Loupy and P. Bellot, "Evaluation of document retrieval systems and query difficulty," in *Proc. Using Eval. HLT Programs Results Trends*, vol. 9, 2000, pp. 32–39.

[35] S. A. Salloum, M. Al-Emran, A. A. Monem, and K. Shaalan, "Using text mining techniques for extracting information from research articles," in *Intelligent Natural Language Processing: Trends and Applications*, vol. 740. Cham, Switzerland: Springer, 2018, pp. 373–397.

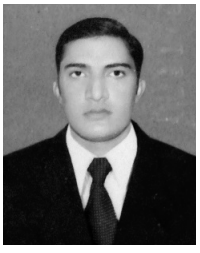
[36] C. de Loupy and P. Bellot, "Evaluation of document retrieval systems," Tech. Rep., 1999.

[37] R. Slavin and J. Niu. (2018). *Security Patterns Repository*. Accessed: Oct. 24, 2018. [Online]. Available: http://sefm.cs.utsa.edu/repository/patterns

[38] J. Kekäläinen and K. Järvelin, "Evaluating information retrieval systems under the challenges of interaction and multidimensional dynamic relevance," in *Proc. 4th CoLIS Conf.*, 2002, pp. 253–270.

[39] (2018). *PDF SAM*. Accessed: Oct. 20, 2018. [Online]. Available: https://pdfsam.org/documentation/

ADNAN KHALID received the Ph.D. degree in cloud computing from the University of Engineering & Technology, Lahore, under the supervision of Prof. Dr. M. Shahbaz. He is currently an Assistant Professor with the prestigious Government College University, Lahore. He teaches research methods and software engineering at the undergraduate and postgraduate level. His area of research is fog computing. He intends to highlight the benefits of this relatively novel field of research. He has four scholarly publications in local HEC recognized journals.



ISHFAQ ALI is currently a Research Scholar with National Textile University, Faisalabad. He is working in the domain of security patterns and their usage in different kind of applications.

MUHAMMAD ASIF received the M.S. and Ph.D. degrees from Asian Institute of Technology, Thailand, in 2009 and 2012, respectively, supported by the HEC foreign Scholarship. He was a Research Scholar with the Computer Science and Information Management Department, Asian Institute of Technology. During the course of time, he was a Visiting Researcher with the National Institute of Information, Tokyo, Japan. He is currently with the Department of Computer Science, National Textile University, Faisalabad. He has worked on some projects, including the Air Traffic Control System of Pakistan Air force. He has authored a number of research papers in reputed journals and conferences. He is a Permanent Member of the Punjab Public Service Commission as an Advisor and a Program Evaluator at the National Computing Education Accreditation Council, Islamabad. He is serving as an Associate Editor for the IEEE Access, the prestigious journal of IEEE. He is also serving as a reviewer of a number of reputed journals.

MUHAMMAD SHAHBAZ received the Ph.D. degree from Loughborough University, U.K. He is currently a Full Professor with the Department of Computer Science and Engineering. He has a wide experience in the field of data science and has published more than 60 papers in the same domain. He has delved several talks in the industry at National and International levels and in various conferences around the world. His research interests include healthier informatics, fog computing, data science, and artificial intelligence.

MARIAM REHMAN received the M.S. and Ph.D. degrees in information management from the Asian Institute of Technology, Thailand. She is currently an Associate Professor with the Department of Information Technology, GC University, Faisalabad, Pakistan. Her research areas include e-government, e-learning, m-learning, databases, information management, and retrieval.

AZIZ GUERGACHI received the Ph.D. degree from the University of Ottawa, Canada. He is currently a Full Professor with the Ted Rogers School of Management, Ryerson University, and an Adjunct Professor with the Department of Mathematics and Statistics, York University, Canada. His research interests include system modeling, explainable AI, and data analytics.