Applied Soft Computing 58 (2017) 225–244

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Software design patterns classification and selection using text categorization approach

Shahid Hussain*, Jacky Keung, Arif Ali Khan

Department of Computer Science, City University of Hong Kong, Hong Kong

ARTICLE INFO

Article history:
Received 3 August 2016
Received in revised form 31 March 2017
Accepted 20 April 2017
Available online 27 April 2017

Keywords:
Deign patterns
Text categorization
Supervised learning
Unsupervised learning
Feature selection
Design problems

ABSTRACT

Context: Numerous software design patterns have been introduced and cataloged either as a canonical or a variant solution to solve a design problem. The existing automatic techniques for design pattern(s) selection aid novice software developers to select the more appropriate design pattern(s) from the list of applicable patterns to solve a design problem in the designing phase of software development life cycle.
Goal: However, the existing automatic techniques are limited to the semi-formal specification, multi-class problem, an adequate sample size to make precise learning and individual classifier training in order to determine a candidate design pattern class and suggest more appropriate pattern(s).
Method: To address these issues, we exploit a text categorization based approach via Fuzzy c-means (unsupervised learning technique) that targets to present a systematic way to group the similar design patterns and suggest the appropriate design pattern(s) to developers related to the specification of a given design problem. We also propose an evaluation model to assess the effectiveness of the proposed approach in the context of several real design problems and design pattern collections. Subsequently, we also propose a new feature selection method Ensemble-IG to overcome the multi-class problem and improve the classification performance of the proposed approach.
Results: The promising experimental results suggest the applicability of the proposed approach in the domain of classification and selection of appropriate design patterns. Subsequently, we also observed the significant improvement in learning precision of the proposed approach through Ensemble-IG.
Conclusion: The proposed approach has four advantages as compared to previous work. First, the semi-formal specification of design patterns is not required as a prerequisite; second, the ground reality of class label assignment is not mandatory; third, lack of classifier's training for each design pattern class and fourth, an adequate sample size is not required to make precise learning.

1. Introduction

Software design is considered as a most challenging task in development life cycle. A number of design methods can be realized in progression process, however, one common category is of architecture-based design methods. This group of design methods uses software architectural styles (presented through components and their relationship) and can be considered as coordination tools among the activities of the software development life cycle. For example, it provides a bridge between satisfactions of system's critical requirements to implementation [1]. Attribute Driven Design (ADD) is a common architecture-based design method which based on an iterative process to apply different architectural strategies and patterns that fulfill quality attributes. In the working procedure of ADD, a software developer selects an architectural style and chooses the desired quality attributes included in the software system. Subsequently, software developers use the description of design requirements related to the selected architectural component and employed software design patterns [2]. Over many years, based on their experiences, software developers have encapsulated and suggested the proven solutions to satisfy the recurring design problems. Subsequently, the experience-based solutions are organized and realize as a standardized form for design patterns. For example, the canonical solution of Composite GoF (Gang-of-Four) design pattern is the result of formulating the recurring structural and compound design problems related to the composition of objects from its nesting and building tree structures. The employment of design patterns in software development is considered as an alternative approach to software refactoring to

* Corresponding author.
E-mail addresses: Shussain7-c@my.cityu.edu.hk (S. Hussain), Jacky.Keung@cityu.edu.hk (J. Keung), aliakhan2-c@my.cityu.edu.hk (A.A. Khan).

http://dx.doi.org/10.1016/j.asoc.2017.04.043
1568-4946/

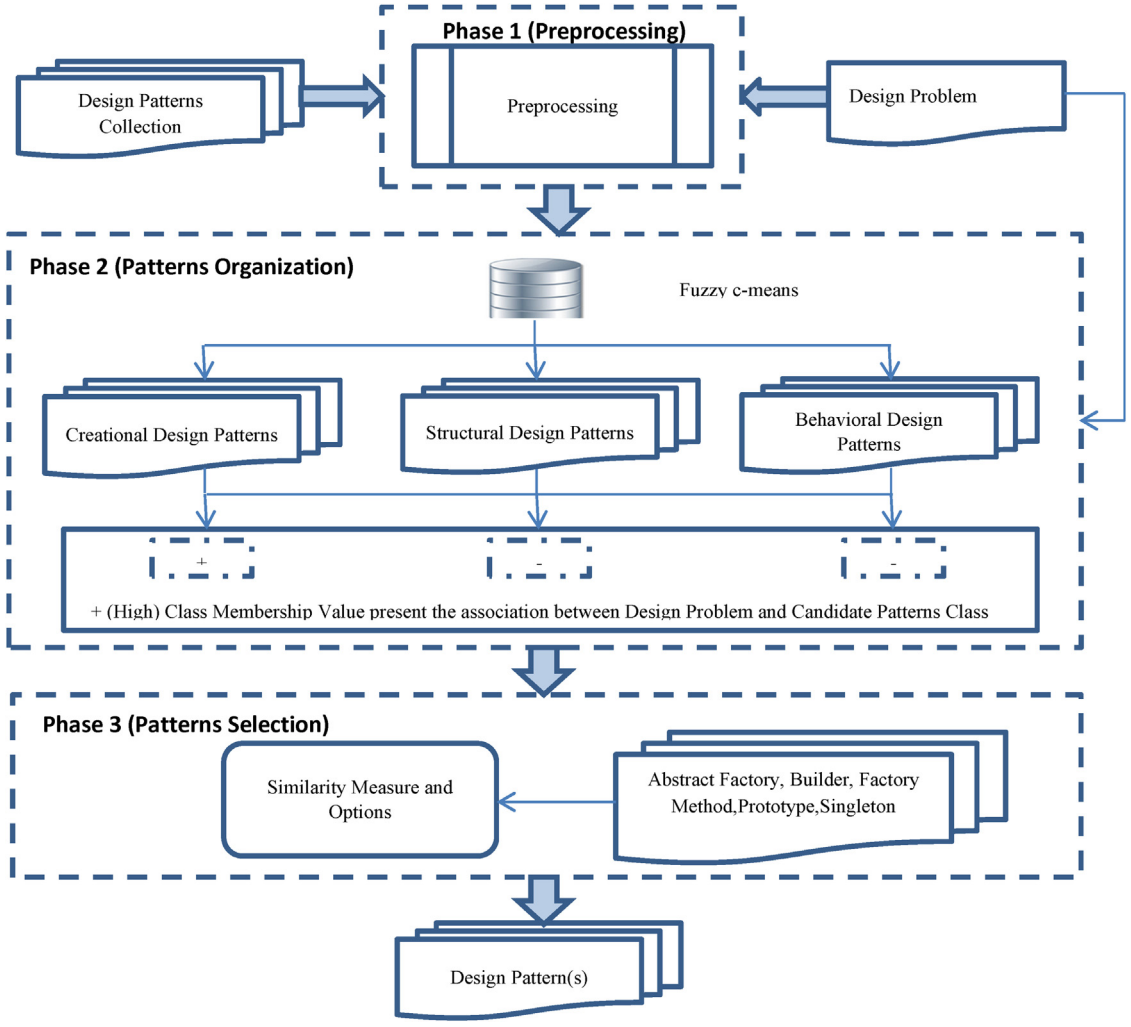


Fig. 1. Overview of the proposed approach.

address the declined quality of software system. Besides, other advantages to employing the design patterns are; to maintain consistency between design and implementation, enabling the relationship between developers, increase the reusability and software modularization [3]. However, due to the existence of spoiled [4], challengeable [5], and interdisciplinary design patterns [6], it is so difficult for a novice designer to find appropriate design pattern(s) and determine its applicability in order to solve a design problem. Since the design patterns are formulated on the base of software developer's experience, consequently, a designer with the lack of experience with design patterns have concerns how to find the best one.

In order to address this issue, we endeavor to automate the suggestion of appropriate design pattern(s) according to an assumed design problem in the design phase of software development. The current efforts to automate the design pattern selection process can be divided into three approaches named UML-Based [7–9], Ontology-Based [10–12] and Text categorization based approach [13]. In the UML-Based approach, the class and collaboration diagrams are used in the analysis phase to describe design patterns and suggest the appropriate one for a design problem. The precise specification of problem definition part of design patterns and meta-model production cost for each design pattern are the main constraints to use the UML-based approach for the selection of appropriate design pattern(s), especially when a designer is dealing with numerous design patterns. Subsequently, in the Ontology-Based approach, different ontologies have been used to define and select the appropriate design pattern(s). This approach is impractical for the automation due to high cost and the existence of similar ontologies in software engineering domains. Finally, in the Text categorization based approach, the ample training sample size and separate classifier for each pattern class (to solve the multi-class problem) are required for the effective results. For example, in the case of complex and interdisciplinary design pattern collection, numerous classifiers are required to make supervised learning for each pattern class [5,6]. The proposed approach is also based on the Text categorization approach [14], aims at unsupervised learning technique Fuzzy c-means to group the similar patterns and select the appropriate design patterns(s) for a given design problem.

The inputs of our proposed approach are the design problem and problem definitions of design patterns of a desired collection. While the existing approaches which have described above uses either semi-formal language (UML) or formal semantics of words used in different ontologies, however, the use of problem definition part of design patterns to determine an appropriate design pattern is highly precise, simple, and less cost effective.

The proposed approach has formulated in three phases (Fig. 1). In the first phase, namely Preprocessing, some text preprocessing activities are performed on the problem definition part of design patterns and design problem before the learning process. Subsequently, in the second phase, namely Patterns Organization, Unsupervised Learning (via Fuzzy c-means) of Design Patterns is

employed for two purposes, 1) group the similar patterns (with or without a design problem description) and referred each group as a design pattern class, and 2) to determine a candidate design pattern class, closer to the description of the given design problem, suggested on the basis of class membership value recommended by Fuzzy c-means. Finally, in the third phase, namely Design Pattern Selection, an appropriate design pattern from the pattern list of the suggested design pattern class is selected for a given design problem. Subsequently, in order to overcome the multi-class problem and to improve learning precision, we also propose a new feature selection method (i.e. Ensemble-IG) using a leveraged scheme IGFSS (Improved Global Feature Selection Scheme) [15]. The proposed method that is the Ensemble-IG (Appendix B) ensemble the one-sided feature selection method OR (Odd Ratio) and global feature selection method Information Gain (IG) regarding the workflow of leveraged scheme IGFSS.

2. Related work

We summarized the related work with respect to efforts made for the classification of design patterns into an appropriate number of design pattern classes and to automate the selection process of the appropriate design pattern(s).

2.1. Design pattern classification

The literature depicts that authors have accomplished different classification schemes to categorize the design patterns on the base of their interest and experience in different domains [3,16–18], for example Rising used the application domain as the category for the classification of design patterns [17]. Usually, authors either use the descriptions of problem's intent or the link between solutions of design patterns in order to divide the design patterns into high-level categories. The problem-based classification schemes, including their domain and high-level categories are summarized in Table 1.

Zimmer considers the different aspects of the link between the established solutions of design patterns, for example, how solutions of two patterns are similar and integrated to each other [22]. Kim and Han analyzed the solution structures of design patterns and consider dependencies to group patterns accordingly [23].

2.2. Design pattern selection

There are three common approaches to frame the researcher's efforts made to automate the selection of design patterns which are UML-Based, Ontology-Based and Text categorization based approaches.

Kim and Khawand depict an approach to select the design pattern by specifying its problem domain. They reported that problem domain of design patterns can help to describe the known problems in the context. The authors used Role-Based Modeling Language (RBML) as UML based pattern specification language to describe the Meta-model to formalize the design patterns. The structure of a Meta-Model is formulated through the class and collaboration diagrams to describe the structural and behavioral aspects of software and detect a design pattern. The 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. Due to the structural similarity between design patterns, a Meta-model cannot specify the problem domain of design patterns. For example, in the case of Gamma et al. design pattern catalog, the Meta-Model for Strategy pattern looks closer to Meta-Model of State design pattern, however, their goals are different. Subsequently, there are some design patterns such as Prototype, whose specification through Meta-Model cannot be presented [7].

In their study, Hsueh et al. also follow the UML-based approach, depict a case study with a limited number of design patterns. Subsequently, the authors consider the four perspectives such as activity-view, requirement-view, problem-view, and tradeoff-view to find a systematic way to select a design pattern. They used UML notations to describe the design problems by incorporating the non-functional requirements besides functional requirements. The design pattern perspectives include, 1) the Activity-View, to define how a design pattern can assist a design activity, 2) Requirement-View, to explore how a design pattern can increase the non-functional requirements, 3) Problem-View, to investigate how a design pattern can help to resolve the design problems and prevent exceptions, and 4) Tradeoff-View, to see how a design pattern can determine the design conflict. The number of limited design patterns in a case study and selection of design pattern through a framework rather than automation are the main constraints [8]. Hasso and Carlson consider the problem definition part of design patterns; perform semantic analysis of sentences and on the base of words meaning to suggest the classification of design patterns. In their study, the authors perform semantic analysis using linguistic theory and consider the meaning of verbs for classification of design patterns. The authors claim that proposed approach helps to select a right design pattern becomes in doubt due to lack of its evaluation and automation [10].

In their study, Khoury et al. introduce an ontological interface depicted in Ontology Web Language (OWL) to aid developers to discover an eligible set of security patterns. The proposed ontological interface implements a mapping between security requirements on one side and security bugs, security errors and threats model on the other side of context [12]. Blomqvist [11] proposed a semi-automatic ontology construction approach named OntoCase, rely on the use of ontology pattern to rank and select the right design pattern. In this work, the author defines criteria which include, relation matching, class matching, density, centrality and semantic similarity to rank the design patterns. The lack of existence of formal description in the software design domains and lack of evaluation reports, are the common constraints to fulfill the aims of these studies. In a recent study, Velasco-Elizondo et al. [24] proposed an approach which based on knowledge representation and information extraction to analyze the architectural pattern description with respect to the specific quality attributes. In order to facilitate the inexperience architects, the proposed approach is automated using a computable model which can be referred as prototype tool. Though, the proposed approach claims the applicability of the proposed approach to focus on the quality attribute-oriented pattern selection and lack of limitation on the use of a pre-defined and closed pattern repository. However, computation time and specification in the case of the complex design pattern collections are the still main issues in the evaluation of the proposed approach [5].

Hasheminejad and Jalili follow the text categorization approach and depicted a two-step automated method to select the design pattern with respect to the degree of similarity between the description of design problems and problem definitions of the retrieved design patterns. The two steps of the proposed method are Learning Design Patterns and Design Pattern Retrieval. In the first step, a classifier is learned for each design pattern class, while in the second step, a candidate design class similar to a design problem is determined based on text categorization. Subsequently, a right design pattern is selected from the design pattern class suggested to the developers. This work has a few shortcomings. First, it needs a classifier for each design pattern class which seems to be inadequate for design pattern catalog, which has been either classified into many categories or obtained as interdisciplinary design patterns [5,6]. Second, the power of classifiers is highly related with sample size and sparse density which are not considered in

Table 1
Summary of the approaches used for problem-based classification of design patterns.

Author	Domain	High-Level Categories
Gamma et al. [3]	Object-Oriented Design Problems	Creational, Structural and Behavioral
Pree [19]	Object-Oriented Design Problems	Pree's classification scheme relies on Recursive Structures and Abstract Coupling
Coad et al. [20]	Object-Oriented Design Problems	Transaction, Aggregate, Plan and Interaction
Tichy [21]	Software Design Problems	Decoupling, Variant Management, State Handling, Control, Virtual Machines, Convenience, Compound, Concurrency and Distribution
Rising [17]	Application Type	Accounting, Hypermedia, Trading, C++ Idioms, System Modeling, Air Defense, Interactive, Database, Persistence and Trading
Douglass [16]	Real Time	Architectural Design, Safety and Reliability, Memory,Concurrency Resources and Distribution
Booch [5]	Architectural Design	45 categories

this work. Third, the authors only use Document Frequency (DF) technique for the feature selection

to evaluate the increase in performance of the proposed method. Though, Document Frequency (DF) is a simple technique, however, it cannot overcome the multi-class problem in feature selection process [13]. In a recent study, Wu et al. [25] proposed a new ensemble method named ForesTexter to overcome the imbalanced text categorization problem and benchmark the results on numerous widely-used imbalanced datasets. Subsequently, authors comparatively investigate the proposed ForesTexter method with standard random forest and different variants of SVM algorithms.

In order to address the shortcomings of previous studies, we proposed an approach, based on Fuzzy c-means unsupervised learning technique and use the problem definition of design patterns to automate the selection of an appropriate design pattern. Subsequently, in order to overcome the multi-class problem and to improve learning precision, we propose a new feature selection method (i.e. Ensemble-IG) by leveraging the workflow of a leveraged scheme IGFSS (Improved Global Feature Selection Scheme) [15].

3. Illustration of the design patterns

Usually, a design pattern template is divided into two sections named Problem Domain and Solution Domain shown in Table 2. The former section defined the context of a problem, while the latter section defined the structure applied to a problem and collaboration with other patterns. The sub-sections of Problem Domain describe the description, scenario illustration of a problem, and a condition required to apply a design pattern. Subsequently, the sub-sections of Solution Domain describe the structure of the participants, its participating components (classes and objects), a collaboration among components and consequences of applied pattern [3]. The illustration of a GoF design pattern named Composite is shown in Table 2.

4. Brief description of text categorization

In machine learning and text mining domains, automatic text categorization is performed either through supervised or unsupervised learning techniques. The former techniques used the class labels which are assigned to documents, and latter techniques used the attributes of the data and dis(similarity) measures to automate their learning. Text preprocessing is mandatory for the text categorization and should perform before learning process [14]. The main steps of text preprocessing are:

4.1. Preprocessing

In the preprocessing step, two activities are performed to transform documents into strings. The first activity is to remove stop words that are more frequent words and carry no information, such as Preposition, Conjunction, and Pronouns. The second activity is word stemming, which is performed to make a group of words that have same conceptual meaning. Subsequently, the numbers of words in the documents collection are reduced. Porter's stemmer is a well-recognized stemming algorithm [26], used to transform English words into their stem iteratively through a set of rules.

4.2. Indexing

In the indexing step, Vector Space Model (VSM) is used as well-known indexing method to describe the documents. The term-document category of VSM is widely used as compared to word-context and pair-pattern matrices categories [27]. Subsequently, for latter categories of VSM (i.e. Word-context and pair-pattern), multi-word terms can be recognized using a domain-independent method [28]. However, in this study, we used the term-document category of VSM, where each document is described by a vector of words. Commonly, a document collection is described through a word-by-document matrix D, where each entry refers to the word's occurrence in the document.

$$D = (W_{wd}) \tag{1}$$

In the Eq. (1), W_{wd} is the weight of word w in document d. There are many ways to determine the weight W_{wd} of word w in document d. For example, Binary, Term Frequency (TF), Term Frequency Inverse Document Frequency (TFIDF), Term Frequency Collection (TFC), Length Term Collection (LTC), and Entropy weighting methods are commonly used in text categorization [14]. The term weighting schemes are used to improve the performance and remove the noise and data sparsity from the documents.

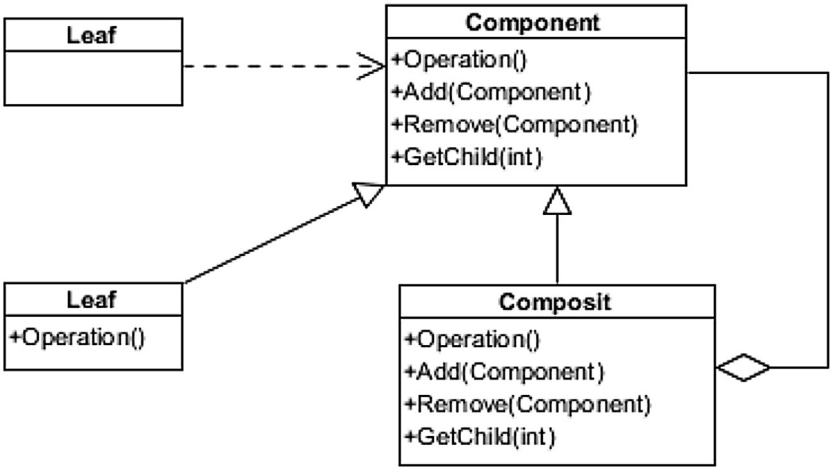
4.3. Feature selection

In the feature selection step, different techniques are used to remove the features. However, the computational time and classification accuracy are the main issues related to the discriminative power of each technique [15]. The wrappers, filters and embedded methods are the three main groups used to categorize the feature selection techniques. The wrapper and embedded techniques are not recommended for the text categorization since these techniques require a frequent interaction of classifiers in their flow, which may increase the running time. Due to its simplicity, efficiency and low running cost, filter-based techniques such as Document Frequency (DF), Information Gain (IG), Mutual Information (MI), Correlation Coefficient (CC), and Chi-Square (CHI) are commonly used in text categorization [14] approach.

5. Overview of proposed approach

The rapid development of software design patterns leads to an increase in the amount of its electronic documentation worldwide. Therefore, there is a need to organize these patterns. This situation increases the importance of text categorization in order to classify the texts into appropriate classes with respect to their con-

Table 2
Illustration of the GoF Composite design pattern.

Design Pattern Name	Composite
Problem Domain	
Intent	"Compose objects into tree structures to represent part-whole hierarchies. Composite lets client treat individual objects and composition of objects uniformly".
Motivation	"Graphics applications like drawing editors and schematic capture system let users build complex diagrams out of simple components. The user can group components to form larger components, which in turn can be grouped to form still larger components. A simple implementation could define classes for graphical primitives such as Text and Lines plus other classes that act as containers for these primitives. But there's a problem with this approach: Code that uses these classes must treat primitives and container objects differently, even if most of the time they use treats them identically. Having to distinguish these objects makes the application more complex. The Composite pattern describes how to use recursive composition so that clients don't have to make this distinction". The rest of motivation section is available in [3]
Applicability	"The composite pattern applies when there is a part-whole hierarchy of objects and a client needs to deal with objects uniformly regardless of the fact that an object might be a leaf or a branch".
Solution Domain	
Structure	
Participant Collaborations	Component, Leaf, Composite and Client.
Consequences	• The Composite Pattern • "Defines class hierarchies consisting of primitive objects and composite objects. Primitive objects can be composed and so on recursively. Whatever client code expects a primitive object, it can also take composite object". • "Makes the client simple. The client can treat composite structure and individual object uniformly. The client normally don't know (and should not care) whether they are dealing with a leaf or composite component. This simplifies client code, because it avoids having to write tag-and-case-statement-style functions over the class that defines the composition". • "Makes it easier to ass new kind of components. Newly defined Composite or leaf subclasses work automatically with existing structure and client code. Clients don't have to be changed for new Component classes". • "Can make your design overall general. The disadvantage of making it easy to add new components is that make it harder to restrict the components of a composite. Sometime you want a composite to have only certain components. With Composite, you can't rely on the type system to enforce those constraints for you. You will have to use run-time checks instead".
Implementation	"The implementation of Composite Design Pattern is available in [3] using C++ constructs".
Related Patterns	"Decorator Pattern: When decorators and composites are used together, they will usually have a common parent class".

tents. There are many domains where text categorization has been applied such as spam e-mail filtering [29], web page classification [30], sentiment analysis [31] and author identification [32]. The literature encourages us to consider the design pattern classification and selection problem as an Information Retrieval (IR) problem. We look at the capacity of automated text categorization in different domains and use in the proposed approach for two purposes, 1) to organize the design patterns via an unsupervised learner (i.e. Fuzzy c-means) and, 2) automatically retrieve a more appropriate design pattern(s) for a given real design problem. We applied the text categorization steps on the description of the Problem Definition part of the design patterns (shown in Table 2) and design problem. We formulate our proposed approach in four steps.

5.1. Preprocessing

The first step of the proposed approach is Preprocessing applied to problem definition of design patterns of the desired collection and given design problem. The sequence of preprocessing activities is revealed in Fig. 2.

The first two activities are performed to remove the stop words and to stem the extracted words. Several text mining tools like JPreText (A simple Text Preprocessing Tool) [33] and their APIs[1] (Application Programming Interfaces) are available to perform these tasks. These two activities also aid to reduce the feature set size and data sparsity. The third activity is performed for index-

[1] http://www.opencalais.com/opencalais-demo/ .

Fig. 2. List of activities performed in the preprocessing phase.

Create	Structure	Support	Client	Abstract	-- -- -- -- --	Class
1	1	0	1	0	-- -- --	1
1	0	1	0	0	-- -- --	0
0	1	0	1	0	-- -- --	1
-	-	-	-	-	-- -- --	-
0	1	1	0	1	-- -- --	1

Fig. 3. Vector Space Model (VSM) for the indexing of GoF Patterns with Binary weighting method.

ing and a Vector Space Model (VSM) includes non-repeated terms of all documents are created. The problem definition part of each design pattern and design problem description is presented in the form of a feature vector. Finally, in the last two activities, one of weighting schemes (Section 4.2) and a feature selection method (Section 4.3) are selected and applied to the vector space model. The last two activities can also aid to analyze the improvement in the efficiency of the proposed approach. In the result of first four preprocessing activities, the structure of Vector Space Model (VSM) (i.e. Eq. (2)) is described with N design pattern and/or a design problem with M non-repeated terms (with binary weights shown in Fig. 3) to assess the performance of the unsupervised learner of the proposed approach. However, the aim of last preprocessing activity is to select the top n features on the base scores assigned through a suggested feature selection method and construct a new VSM. Subsequently, the impact of feature selection methods on the performance of the unsupervised learner is evaluated according to new constructed VSM. In this paper, one of our contributions is to introduce a new feature selection technique named Ensemble-IG by leveraging IGFSS (Improved Global Feature Selection Scheme) [15]. The proposed method, namely Ensemble-IG, ensemble the one-sided feature selection method OR (Odd Ratio) and global feature selection method Information Gain (IG) regarding the workflow of leveraged scheme IGFSS. The procedure used to describe the flow of Ensemble-IG is shown in Appendix B. Subsequently, we evaluate the Ensemble-IG with other feature selection techniques used in this study.

$$\mathrm{VSM} = \sum_{i=1}^{N}\sum_{j=1}^{M} w(T_{i,j}) \tag{2}$$

5.2. Unsupervised learning via fuzzy c-means for design patterns

Though there are several unsupervised learning methods, however, it cannot be said which one is better than others [34]. For example, partitioning (K-means, K-medoids), hierarchical (Agglomerative and Divisive), model-based (Decision Tree, Neural Network), grid-based, density-based (DBSCAN, AUTOCLASS) and soft-computing (Fuzzy c-means) [14]. However, the main purpose of the proposed approach is to use Fuzzy c-means, an unsupervised learner in order to classify the design patterns with or without prior knowledge of their class labels [35]. The aim of Fuzzy c-means is to classify the objects (i.e. Design Patterns in our study) into clusters (i.e. Design pattern classes) according to their membership degree values between 0 and 1. Like other clustering techniques, the FCM uses some notations to describe the functionality of the algorithm. We have described the functionality of Fuzzy c-means in four stages as follows.

Stage 1. (Description of Dataset Structure)

In the first stage of Fuzzy C-means, the dataset with N design patterns and M features, M-dimensional column vectors, and N observations can be represented through the Eqs. (3a)–(3c) respectively.

$$X = \begin{bmatrix} x_{11} & x_{12} & - & x_{1M} \\ x_{21} & - & - & - \\ - & - & - & - \\ x_{N1} & x_{N2} & - & x_{NM} \end{bmatrix} \tag{3a}$$

$$x_k = [x_{1k}, x_{2k}, \ldots, x_{Nk}]^T, x_k \in \mathbb{R}^n \tag{3b}$$

$$X = \{x_k \mid k = 1, 2, \ldots, M\} \tag{3c}$$

Stage 2. (Implementation of Objective function)

In the second stage, the aim of the Fuzzy c-means algorithm that is the minimization of the objective function of Fuzzy c-means algorithm is described through the Eq. (4a).

$$J_m(X; U, V) = \sum_{t=1}^{c}\sum_{k=1}^{M} (U_{tk})^m \|x_k - v_t\|_A^2 \tag{4a}$$

Where U and V are the Fuzzy c-partition and vectors for centers of the dataset X and are described through the Eqs. (4a)–(4c) respectively. The term c is referred as the number of clusters for X. Subsequently, weighting component (i.e. m) and weight Matrix (i.e. A) are the two main parameters of the objective function J_m. The aim of m (i.e. $1 \leq m < \infty$) is to control the relative weights which are used for the squared error D_{tkA}^2 (Eq. (5)). The increase in the value of m incline to reduce the membership for the fuzziest state. Though the better value of m can be derived through an experimental investigation, however, usually $1.5 \leq m \leq 3.0$ is used to produce good results. The second parameter of the objective function J_m of Fuzzy c-means is the weight matrix A, which controls the cluster's shape in $\mathbb{R}^n$. Though there are several A-norms such as Euclidean (A = I), Diagonal ($A = D_y^{-1}$), and Mahalonobis ($A = C_y^{-1}$) Norms are available and

each norm on $\mathbb{R}^n$ can be computed using the Eq. (5). The relationship of data sparsity with distance metrics and their impact on the clustering is a controversial issue in the research community[2] [36–40]. However, the use of Euclidean norm remained the better choice. In the proposed approach, we employed the Fuzzy c-means with the Euclidean norm and tune m with $1.5 \leq m \leq 3.0$.

$$U = [\mu_{tk}] \in M_{fc} \tag{4b}$$

$$V = [v_1, v_2, \ldots, v_c],\ v_i \in \mathbb{R}^n \tag{4c}$$

Stage 3. (Computation of squared inner-product distance norm)

The squared distance between x_k and v_i is computed in the A-norm via the Eq. (5). The Fuzzy c-means of the proposed approach with Euclidean distance norm aid to compute the closeness between data objects (i.e. Design patterns in our study).

$$D^2_{tkA} = ||x_k - v_i||^2_A (x_k - v_i)^T A (x_k - v_i) \tag{5}$$

Stage 4. (Identifying the optimality in Fuzzy c-partition)

Though numerous clustering criteria have been used for identification of optimal Fuzzy c-partitions for X, however widely used to generalized the least-squared error $(U_{tk})^m$. In the Eq. (5), the optimality in the Fuzzy c-means for the X can be described through the pairs (U_i, v_i) for cluster i, which can locally minimize the J_m (as objective function) using the Eqs. (6a) and (6b).

$$V_i = \frac{\sum_{k=1}^{M} (U_{ik})^m x_K}{\sum_{k=1}^{M} (U_{ik})^m};\ 1 \leq i \leq c \tag{6a}$$

$$U_{ik} = \left(\sum_{t=1}^{c} \left(\frac{d_{ik}}{d_{tk}} \right)^{2/m-1} \right)^{-1};\ 1 \leq k \leq M,\ 1 \leq i \leq c \tag{6b}$$

The entropy (Eq. (6c)) and partition coefficient (Eq. (6d)) are widely used to measure the optimality of Fuzzy c-means by minimizing the J_m globally.

$$H_c(U) = -\sum_{t=1}^{c} \sum_{k=1}^{M} (U_{tk} \log(U_{tk}))/M \tag{6c}$$

$$F(U) = \sum_{t=1}^{c} \sum_{k=1}^{M} (U_{tk})^2 \tag{6d}$$

5.3. Identification of design pattern class

In the proposed approach, a design pattern class referred as a group of suggested patterns for a given design problem. The number of design pattern classes depends on the decision of Fuzzy c-means used in the proposed approach. For example, in Fig. 1, we choose Gang-of-Four design pattern collection to illustrate the design pattern classes, however, for other design pattern groups, the number of corresponding classes can be varied (Section 6.3). The design pattern class with the highest membership value (recommended by Fuzzy c-means) is considered as candidate design pattern class for a given design problem and represent with a + symbol. The highest class membership value depicts more closeness between the textual similarities for problem definition part of design patterns (i.e. Members) of corresponding candidate class (i.e. Clusters) and the description of the design problem.

5.4. Design patterns suggestion and selection

The candidate design pattern class for the given design problem is identified by Fuzzy c-means (Section 5.2) depict the list of relevant patterns. Subsequently, in order to recognize the most appropriate design pattern(s) for a given problem, similarity measures such as Cosine, Pearson Correlation, Extended Jaccard and Dice Coefficient are used [41]. However, we used well-known Cosine Similarity (CS), since it aid to measure the correlation between two vectors and is independent of document length. The correlation between design problem and design patterns of the candidate class (recommended by the unsupervised learner) is calculated based on Eqs. (7)–(8).

$$CS_i = \sum_{j=1}^{N} w(\text{Pattern}_i, t_j) \times w(\text{Problem}, t_j) \tag{7}$$

$$W(d) = (w(d,t_1), w(d,t_2), \ldots, w(d,t_n)) \tag{8}$$

In the Eq. (7), the Pattern_i and Problem are the problem definition of ^{i}th design pattern and description of design problem respectively. The term weighted vector for Pattern_i and Problem is described in through the Eq. (8). After finding similarities between design problem and design patterns of candidate class using the Eq. (7), one of the following Eqs. (9)–(11) can be applied to suggest design pattern(s) to a developer.

5.4.1. Suggestion of best design pattern

The Eq. (9) can suggest k^{th} design pattern (with highest CS_i value) to the developer.

$$k = \arg\max_i CS_i \tag{9}$$

5.4.2. Suggestion of design patterns with respect to a threshold $\theta 1$

The Eq. (10) can suggest zero, one or more than one design patterns with CS_i to the developer, depends on the value of $\theta 1$.

$$|CS_i| > \Theta_1 \tag{10}$$

5.4.3. Suggestion of design patterns with respect to a threshold $\theta 2$

The Eq. (11) can suggest design patterns contiguous to the best design pattern (with CS_{max}) to the developer.

$$|CS_{max} - CS_i| \leq \Theta_2 \tag{11}$$

The efficient threshold values for θ_1 and θ_2 can be obtained from the experiment according to performance evaluation of proposed approach [13,42].

6. Evaluation model for proposed approach

The evaluation model for the proposed approach is performed in three steps. The first step formulates the evaluation of unsupervised learning technique that is Fuzzy c-means to determine the candidate design pattern class for a particular design problem. Subsequently, the second step formulates the evaluation of proposed options describe in Section 5.4 to suggest appropriate design pattern(s) from the candidate design pattern class. Finally, the third step formulates the use of resources includes three design pattern collection and 80 real design problems in order to evaluate the proposed approach.

[2] https://www.researchgate.net/post/What_is_the_best_distance_measure_for_high_dimensional_data .

6.1. Evaluation of unsupervised learning technique (Fuzzy c-means)

The evaluation performance of unsupervised learning techniques is not as trivial as we consider precision and recall of a supervised learning technique. However, some ground truth set of classes or some assumptions which satisfied that members belong to the class is more similar using some sort of similarity metrics. For example, the Adjusted Rand Index (ARI), V-measure and Mutual Information (MI) metrics need ground truth set of classes to evaluate unsupervised learning techniques [43]. However, Silhouette Coefficient metric did not need ground truth labels and learning technique itself perform evaluation [44]. The effectiveness of Fuzzy c-means depends on the highest value of any one of these similarities ' metrics. Subsequently, in this study, we used Adjusted Rand Index (ARI) measure to evaluate learning precision of Fuzzy c-means. We consider three design pattern collections (are already categorized by some groups of software engineering experts [3,16,45]) and 80 real design problems evaluate the of classification effectiveness of the proposed approach.

Therefore, the classification effectiveness of Fuzzy c-means is measured in three steps. In the first step, the performance measures Recall (R), Precision (P), and F-measure aid to select the best weighting method for the Fuzzy c-means learner. Subsequently, in the second step, Adjusted Rand Index (ARI) similarity metric aid in evaluating the classification effectiveness of Fuzzy c-means. Finally, in the third step, Document Frequency (DF), Correlation, Information Gain (IG), Chi-Square and Gain Ratio (GR) [46] are used in the evaluation model. Subsequently, in order to overcome the multiclass problem and to improve learning precision, we also propose a new feature selection method (i.e. Ensemble-IG) using a leveraged scheme IGFSS (Improved Global Feature Selection Scheme) [15] and ensemble with Information Gain (IG).

In the text categorization approach, Precision and Recall for the learners can be estimated using micro-averaging or macro-averaging equations, however, the precision of micro-averaging (used in this study) is better than macro-averaging. In order to select the best weighting method for Fuzzy c-means learning technique, the effect of each method such as Binary, TFIDF, TF, TFC and Entropy on Fuzzy c-means is computed in term of Precision, Recall and F-measure [15,46]. Subsequently, weighting method with highest F-measure values is chosen for the Fuzzy c-means.

$$P = \frac{\sum_{c=1}^{C} TP_c}{\sum_{c=1}^{C} (TP_c + FP_c)}, \text{Micro-Averaging} \quad (12)$$

$$R = \frac{\sum_{c=1}^{C} TP_c}{\sum_{c=1}^{C} (TP_c + FN_c)}, \text{Micro-Averaging} \quad (13)$$

$$F = \frac{2 \times P \times R}{(P+R)} \quad (14)$$

In Eqs. (12)–(13), C is the number of design pattern classes decided to evaluate the performance of learning techniques used in the proposed approach. For example, in the case of GoF design pattern collection, the value of C is 3 (Section 6.3). The term TP is the number of design problems which are correctly identified for each design pattern class, FP is the number of design problems which are incorrectly identified for each design pattern class, and FN is the number of design problems which are missed from each corresponding design pattern class. The values of P, R, and F are computed using Eqs. (12)–(14) respectively.

The value of ARI is computed using Eqs. (15), (16) and an unsupervised learner recommends N (number of design pattern classes) with the highest ARI value.

$$ARI = \frac{\text{RandomIndex - E [RandomIndex]}}{\text{max(RandomIndex) - E[RandomIndex]}} \quad (15)$$

$$\text{RandomIndex} = \frac{P_a + P_b}{Cl_2^{n_{samples}}} \quad (16)$$

In Eqs. (15), (16), Cl is a ground truth class label and K is the assumed number of classes decided by the unsupervised learner. Suppose we have two partitions P_1 (with n_1 elements) and P_2 (with n_2 elements) for n elements (in the case of GoF pattern group, the value of n will 23) with the same number of classes C (i.e. In case of GoF pattern group, the value of C will 3). Subsequently, we suppose that C_1 and C_2 are the two disjunctive tables and T as a contingency table with n_{ij} elements. In order to compute the Random Index, each partition P is characterized by $n \times n$ paired comparison table to evaluate the total pair of agreements. For example.

a) Pairs belong to the same classes of both P_1 and P_2.

b) Pairs belong to different classes of both P_1 and P_2.

c) Pairs belong to the same class of P_1 but to different classes of P_2.

d) Pairs belong to the same class of P_2 but to different classes of P_1.

The first two cases (a & b) referred as a total number of agreements while the last two cases (c & d) referred as the total of discordances. The term (P_a in Eq. (16)) is the number of pairs of elements that are identified in the same set of Cl and K, while P_b is the number of pairs of elements that are identified in the different set of Cl and K [47,48].

6.2. Evaluation of selection of design pattern(s) process

After identification of a candidate design pattern class, next step is to select more appropriate design pattern(s) for a given design problem. In this step of evaluation, similarity equations described in Section 5.4 are applied and more appropriate design pattern(s) are suggested to the developer. The suggested design pattern(s) are retrieved from the list of candidate design pattern class. The ratio of correctly suggested design patterns and total suggested design pattern, aid in choosing the best similarity equation. The similarity equation with the highest value of RCD (Ratio of Corrected Design patterns), computed by Eq. (17), is selected.

$$RCD = \frac{\text{Number of Right Suggested Design Pattern}}{\text{Total Suggested Design Patterns}} \quad (17)$$

6.3. Design pattern groups

In software engineering, some groups of experts from different domains have categorized the design patterns on the base their intent, motivation, and applicability. We consider three frequently used design pattern collection in our proposed evaluation model.

6.3.1. Gang-of-Four (GoF) design pattern collection

The GoF collection includes 23 object-oriented design patterns which are divided into three groups named Creational, Structural, and Behavioral. This case study includes 1465 non-repeated words of all 23 design patterns after removing the stop words and words stemming [3].

6.3.2. Douglass design pattern collection

The Douglass collection includes 34 real time system relevant design patterns which are divided into five categories named Concurrency, Safety, and Reliability, Distribution, Memory, and Resource. This case study includes 1271 non-repeated words of all 34 design patterns after removing the stop words and words stemming [16].

6.3.3. Security design pattern collection

The Security collection includes 46 security systems relevant design patterns which are divided into eight categories named SACA (System Access and Control Architecture), ACM (Access Control Model), IA (Identification and Authentication), OSAC (Operating System Access Control), SIA (Security Internet Application), FA (Firewall Architecture), ESRM (Enterprise Security and Risk Management) and Accounting [45]. This case study includes 1230 non-repeated words of all 34 design patterns after removing the stop words and words stemming.

6.4. Real design problems

In order to evaluate the effectiveness of proposed framework, we retrieve 80 real design problems from different resource [4,18,45,49–51]. We retrieved the design problems which have been addressed and mapped according to expert opinions, and aid us to investigate the authenticity of the proposed framework. The reference of resources, authors who address and mapped the problems, and the involvement of the community is shown in Table 3.

In this section, we only provide the description of nine design problems (three for each pattern collection defined in section 6.3) to evaluate the effectiveness of the proposed method.

6.4.1. Problems for GoF patterns collection

Design Problem 1: "Design a system enabling to display on a screen some empty windows (no button, no menu. . .). A window can have several different styles depending on the platform used. We consider two platforms, XWindow and Presentation Manager. The client code must be written independently and without knowledge of the future execution platform. It is probable that the system evolves in order to display specialized windows by 'application windows' (able to manage applications) and 'iconised windows' (with an icon)" [4].

Design Problem 2: "Design the communications of one plane to the approach of an airport. When a plane is in approach of the airport, it must announce to all the other planes which are around that it intends to be posed, and await their confirmation with all before carrying out the operation. It is the control tower of the airport which guarantees the regulation of the air traffic, by making sure that there is no trajectory conflict or destination between several planes. Besides the class diagram, represent by a collaboration (diagram of collaboration or diagram of objects and sequence) the landing of a plane among two wanting to land and one wanting to take off." [4].

Design Problem 3: "Many distinct and unrelated operations need to be performed on node objects in a heterogeneous aggregate structure. You want to avoid 'polluting' the node classes with these operations. And, you don't want to have to query the type of each node and cast the pointer to the correct type before performing the desired operation." [51].

6.4.2. Problems for douglass pattern collection

Design Problem 4: "How to protect the programs from one another and the kernel from them all and how to handle relocation in case, when more than one program reside in main memory, waiting either for I/O (Input/Output) or CPU resource, for better CPU utilization" [18].

Design Problem 5: "In case of limited resources such as I/O devices, how process should be model to access these resources without any deadlock and starvation. For example, how process (philosophers) should be model in dining philosophers problem, who try to access limited number of resources (i.e. fork)." [18].

Design Problem 6: "A distributed system using the mailboxes has two IPC primitives, send and receive. The latter primitive specifics a process to receive from and blocks if no message from that process is available, even though message may be waiting from other process. Is deadlock possible, if there are no shared resources, but process need to communicate frequently" [18].

6.4.3. Problems for security pattern collection

Design Problem 7: "The Role-based access control model is used now in many systems. However, the different component frameworks (.NET, J2EE) provide support only to define roles and to write authorization rules, and do not say anything about where the rights come from. It is not easy for system designers or for administrators to define the required roles and their corresponding rights. How can we assign appropriate rights to the roles when we want to implement a least privilege policy?"[45].

Design Problem 8: "Audit is a security service that scrutinizes logs, audit trails or other captured information and attempts to discern more detailed information about an event. It analyzes the event information for any indication of security violations. You need a clear set of requirements to ensure that the audit strategy employed actually satisfies the needs of the organization or system. Requirements for audit often conflict with each other, and trade-offs among them are often necessary. How can you determine a balanced set of specific requirements for an audit service, and their relative importance? [45].

Design Problem 9: 'The ability to define an asset's value is a key component of any risk assessment. Threats and vulnerabilities that target and expose an asset are only significant within the context of the asset's value. Without this determination, an enterprise is unable to properly assess the risks posed to its assets. How can an enterprise determine the overall value of its assets?' [45].

7. Pseudocodes to describe the implementation procedure of the proposed approach

7.1. Organization of design patterns

In this section, we presented three pseudocodes to describe the experimental procedures used for the implementation of the proposed approach. The aim of pseudocodes is to describe the procedure that how the proposed approach can be employed to; 1) organize the design patterns of a collection, 2) determine an appropriate pattern class for a given design problem, and 3) select the right design pattern(s) from a suggested pattern class. The pseudocode-1 describes the procedure for the organization of design patterns of a collection as follows.

Pseudocode-1 Pattern_Organization (PC, k, N, M)

Input

PC = {Set of problem domain description of design patterns in the given pattern collection}

k = The number of clusters to organize the pattern catalog, which is recommended through an expert opinion, For example, in the case of GoF (k = 3) and Douglass (k = 5).

N = The number of design patterns in a collection, For example, in the case of GoF and Douglass pattern collection, the value of N is 23 and 34 respectively.

M = The number of unique words, For example, in the case of GoF and Douglass pattern collection, the value of M is 1465 and 1271 respectively.

Table 3
Resource information about design problems.

Reference of Resoure	Authors	Involved Participants	Related Pattern Collection
Bouhour et al. [4]	C. Bouhour, H. Leblance, and C. Percebois	126	Gang-of-Four patterns
Silberschatz et al. [18]	A. Silberschatz, P.B. Galvin, and G. Gagne	3	Real Time Design Patterns
Schumacher et al. [45]	M. Schumacher, E. F. Buglioni, D. Hyberston, F. Buschmann, P. Sommerlad	5	Security Patterns
Tanenbaum et al. [49]	A.S Tanenbaum and H. Bos	2	Real Time Design Patterns
Shalloway et al. [50]	A. Shalloway and R. Trott	3	Gang-of-Four patterns
Shvets [51]	A. Shvets	1	Gang-of-Four patterns

Procedure Begin

- Perform first three activities of preprocessing phase of the proposed approach to construct a Vector Space Model (VSM) with N design patterns and M Unique words by tuning the data sparsity level.
- Apply a weighting method (such as Binary, TF, TFIDF, TFC, LTC, and Entropy) to a constructed VSM.
- Apply the unsupervised learning technique (Fuzzy c-means with Euclidean distance measure) to organize the N design patterns.
- Apply the evaluation criteria (F-measure in Section 6.1) to assess the performance of the Fuzzy c-means with corresponding weighting method and k clusters.
- Repeat Steps 2, 3 and 4 until the all possible assessments have done.
- Select the best weighting method for fuzzy c-means with highest F-measure value.
- In order to evaluate the effectiveness and robustness [52] of Fuzzy c-means at k (e.g. k = 3 for GoF pattern collection), we compute the performance of Fuzzy c-means with best weighting method and with the different number of clusters (For example, the value of k varies between 2–10). Subsequently, we used the ARI (i.e. The Eqs. (15) and (16)) to evaluate the degree of closeness with predicted and ground truth class labels for patterns.
- In order to evaluate the improvement in the effectiveness of Fuzzy c-means, apply a feature selection method (Such as Document Frequency (DF), Information Gain (IG), Mutual Information (MI), Correlation Coefficient (CC), Chi-Square (CHI), and Ensemble-IG) to rank the top n features.
- The performance of fuzzy c-means is assessed in terms of F-measure on the top n features (i.e. The value of n varies from 10 to 350).
- The improvement in the performance of Fuzzy c-means with M (i.e. Feature set with actual M features, such as in Step 6) and n (i.e. Feature set with n features, such as in Step 9) features could be compared.

Procedure End
Output:

The Fuzzy c-means with best weighting and feature selection method for the organization of design patterns catalog (i.e. PC).

The objective of pseudocode-1 is to provide guidelines regarding three activities, 1) to select the best weighting method for Fuzzy c-means with respect to a target design pattern collection, 2) to evaluate the effectiveness of Fuzzy c-means at k according to expert opinion, for example, k = 3 in the case of GoF pattern collection, and 3) to investigate the impact of feature selection methods on the performance of Fuzzy c-means.

Subsequently, though we have given the description of 9 out of 80 design problems which we considered from different resources [4,18,45,49–51] (Section 6.4). For each design problem, the effectiveness of the proposed approach is evaluated by determining the right design pattern class. The pseudocode-2 describes the procedure for the organization of design patterns of a collection with respect a related design problem using the proposed approach.

Pseudocode-2 Determing_Pattern_Class_For_Problem (PC, D, k, N, M)
Input:

PC = {Set of problem domain description of design patterns in the given design pattern Collection}

DP = {Set of descriptions of the given design problems described in the context of target pattern collection}

k = The number of clusters to organize the pattern catalog, which is recommended through an expert opinion, For example, in the case of GoF (k = 3) and Douglass (k = 5).

N = The number of design patterns in a collection and a design problem, For example, in the case of GoF N will equal to 24 (i.e. 23 design patterns and 1 design problem)

M = The number of unique words

Procedure Begin

- Select a design problem from the DP
- Perform first three activities of preprocessing phase of the proposed approach to construct a Vector Space Model (VSM) with N + 1 (i.e. N design patterns and 1 selected design problem) feature vectors and M unique words by tuning the data sparsity level.
- Apply a weighting method (such as Binary, TF, TFIDF, TFC, LTC, and Entropy) to a constructed VSM.
- Apply the unsupervised learning technique (Fuzzy c-means with Euclidean distance measure) to organize the N design patterns and 1 selected design problem.
- Apply the evaluation criteria (F-measure in Section 6.1) to assess the performance of the Fuzzy c-means with corresponding weighting method and k clusters.
- Repeat Steps 3, 4 and 5 until the all possible assessments have done.
- Evaluate the performance of Fuzzy c-means with the best weighting to determine the right design pattern class for the selected design problem.
- Repeat Step 1 to Step 7 until the pattern class for each design problem of DP is determined.

Procedure End
Output

The suggested pattern class (i.e. Category) for the given sample of design problems (DP) via Fuzzy c-means of the proposed approach.

7.2. Selection of right design pattern(s)

In Section 7.1, we have described the procedure to organize the design patterns of a collection and suggestion of right pattern class for the given design problems via the unsupervised learner of the proposed approach. However, in this section, we described the procedure that how the right pattern(s) can be selected from the list of patterns of suggested pattern class.

Pseudocode-3 Pattern_Selection_For_Problem (Pattern_Vectors, Problem_Vector)
Input:

Pattern_Vectors = {Set of the feature vectors for design patterns of a pattern class recommended for the target design problem}

Problem_Vector = Describe the feature vector for target design problem

Procedure Begin

- The Eq. (7) (cosine similarity measure) can be used to measure the closeness of patterns with the given design problems.
- The similarity options are assessed to select the right design pattern(s).
  - ○ The Eq. (9) described the first similarity option (Section 5.4.1) which aids to recommend the right design pattern for a given design problem on the base of cosine similarity value. For example, in the case of Design Problem 1 (Section 6.4), the Bridge is a recommended pattern with highest cosine similarity value (i.e. 0.51) as compared to other patterns of the same class.
- The Eq. (10) described the second similarity option (Section 5.4.2) which aids to recommend zero, one or more than design patterns for a given design problem based on the suggested threshold (Θ_1) value, which might be an average of the cosine similarity values. For example, in the case of Design Problem 1 (Section 6.4), the suggested threshold value is 0.34–0.50 (Table 7) to recommend the zero, one or more than one design pattern(s). The number of recommended patterns depends on the selected value of the suggested threshold range. For example, in the case of Θ_1(less than 0.34) more than one right design pattern can be recommended. Subsequently, in the case of $\Theta_1$greater than 0.51, no pattern is recommended.
- The Eq. (11) described the third similarity option (Section 5.4.3) which aids to recommend design patterns contiguous to the best design pattern (with CS_{max}) based on the suggested threshold (Θ_2) for cosine similarity values. For example, in the case of Design Problem 1 (Section 6.4), the suggested threshold value is 0.0–0.16 (Table 7). The number of suggested patterns depends on the selected value from the suggested range of threshold (Θ_2).
- Finally, the Eq. (17) is used to select the best similarity option (Section 5.4) by considering the ratio of the number of right suggested and total suggested design patterns. For example, on the sample of nine design problems (Section 6.4), the first similarity option (from Eq. (9)) with highest RCD (Ratio of Corrected Design patterns) is recommended.

Procedure End

Output

The recommendation of right design pattern(s) from the patterns list (Pattern_Vectors) of determined pattern class with respect to content similarity for the given design problem (i.e. Problem_Vector) and best similarity option is suggested.

7.3. Used tool in experiments

We performed all experiments on Intel® Core™ i7-2600 at 3.40 GHz with 8192MB RAM. We used R Project for Statistical Computing (R-3.3.3) in the context of Window 7. Subsequently, We used the "tm", "worldcloud", "snowballC", and "e1071" R packages to perform the experiments according to the procedures described through the pseudocodes in Section 7.1. The source for R-3.3.3 and related packages is the internet.

8. Results and discussion on evaluation of proposed approach

The proposed approach is evaluated according to the evaluation model described in Section 6. In order to improve the homogeneity and completeness features, the most appropriate weighting method (in term of F-measure) has been used for the Fuzzy c-means unsupervised learning technique of the proposed approach. However, weighting method can be explicitly revealed. The unsupervised learning technique Fuzzy c-means with its best weighting method suggest N design pattern classes with the highest ARI value.

8.1. Evaluation of learning for design pattern collections

In order to determine the right number of design pattern classes with respect to ground reality, the effectiveness of unsupervised learning technique Fuzzy c-means is evaluated using the three design pattern collection and evaluation model described in the Section 6.3 and 6.1 respectively. In each evaluation process, Binary, TFIDF, TFC, LTC, and Entropy weighting methods are applied; however, the effectiveness of learning technique is evaluated with the best weighting method.

8.1.1. Gang-of-Four (GoF) design patterns collection evaluation

The patterns in GoF collection are divided into three categories. According to F-measure criterion, the evaluation result in the selection of best weighting method for the Fuzzy c-means is shown in Fig. 4-a. For example, TDIDF with highest F-measure is suggested as a best weighting method for the Fuzzy c-means. While Fig. 5-a depict the evaluation results of Fuzzy c-means (with best weighting method) with a different number of clusters k (each cluster is considered as pattern class). According to ARI criterion, Fuzzy c-means recommend a number of design pattern classes (k = 3 in Fig. 5-a) with highest ARI, which is correct according to expert's opinion [3]. The classification of design patterns into an appropriate number of design pattern classes suggests the applicability of the proposed approach.

We have incorporated feature selection activity in the preprocessing step of the proposed approach (Section 5.1), in order to increase the classification performance by reducing the dimension of feature vectors. Therefore, we applied six feature selection methods Chi-Square, Document Frequency (DF), Correlation, Information Gain, Gain Ratio including our proposed technique named Ensemble-IG (to overcome the multi-class problem) and evaluate their impact on the performance of Fuzzy c-means. The evaluation results are shown in Fig. 5-b. The x-axis of Fig. 5-b (labeled with Number of Rank Features) refers to top n features rank by each feature selection method.

According to F-measure criterion, the feature selection method with the highest increase in the performance of Fuzzy c-means as compared to its actual performance F-measure = 0.73 before feature selection (illustrated in Fig. 4-a) is suggested. For example, with top 10 features (depicted on the x-axis of Fig. 5-b) rank by each feature selection method, there is an increase from 6.84% to 15.06% (in term of F-measure) in the performance of Fuzzy c-means. However, the proposed feature selection method Ensemble-IG with the highest increase in the performance (i.e. 15.06%) is suggested for the fuzzy c-means. We have replicated this procedure with Top n (where n ranges from 50 to 350) features and illustrated the evaluation results in Fig. 5-b.

8.1.2. Douglass design patterns collection evaluation

The patterns in Douglass collection are divided into five categories. According to F-measure criterion, the evaluation result in the selection of best weighting method for the Fuzzy c-means used in proposed approach is shown in Fig. 4-b. For example, TDIDF with

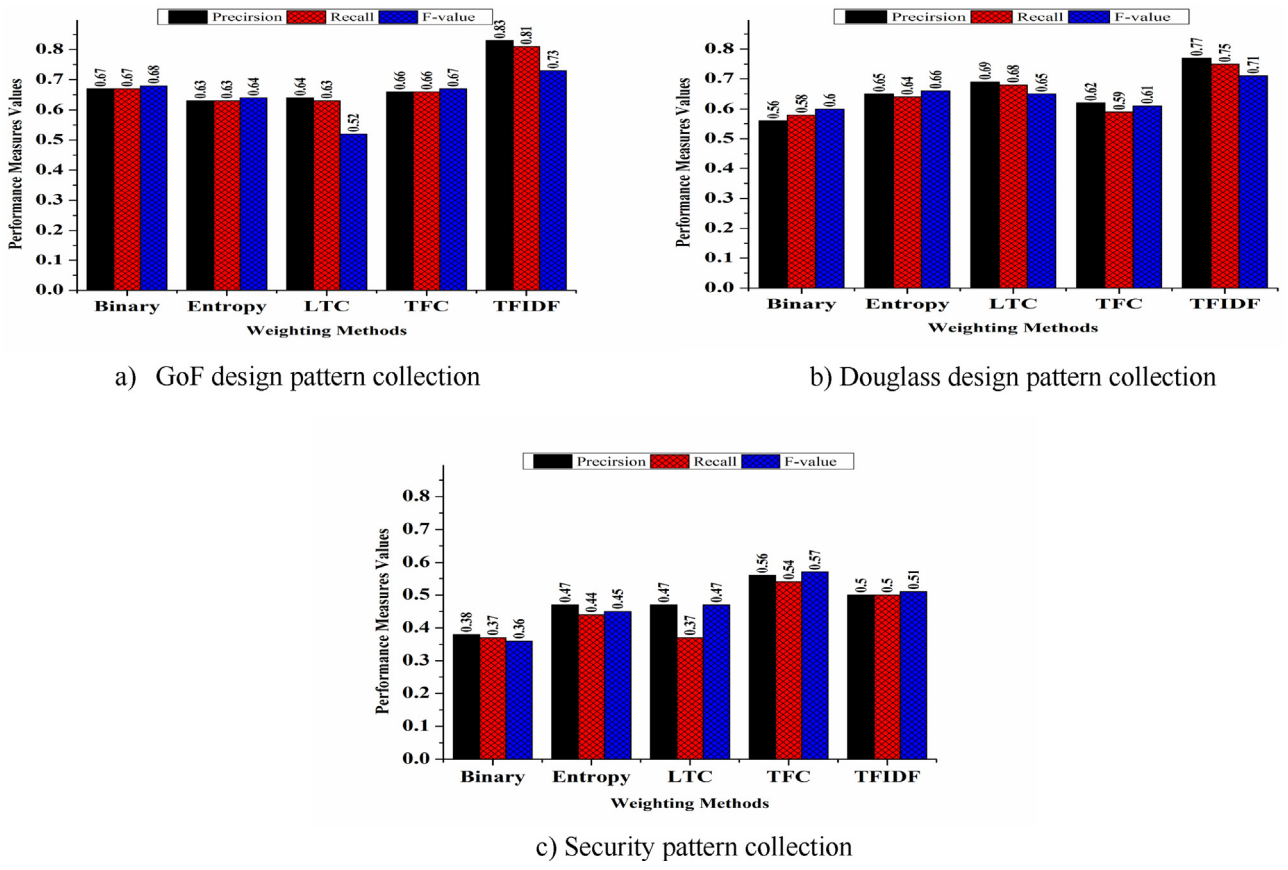


Fig. 4. Best weighting methods for Fuzzy c-means on. a) GoF design pattern b) Douglass design pattern c) Security design pattern collections

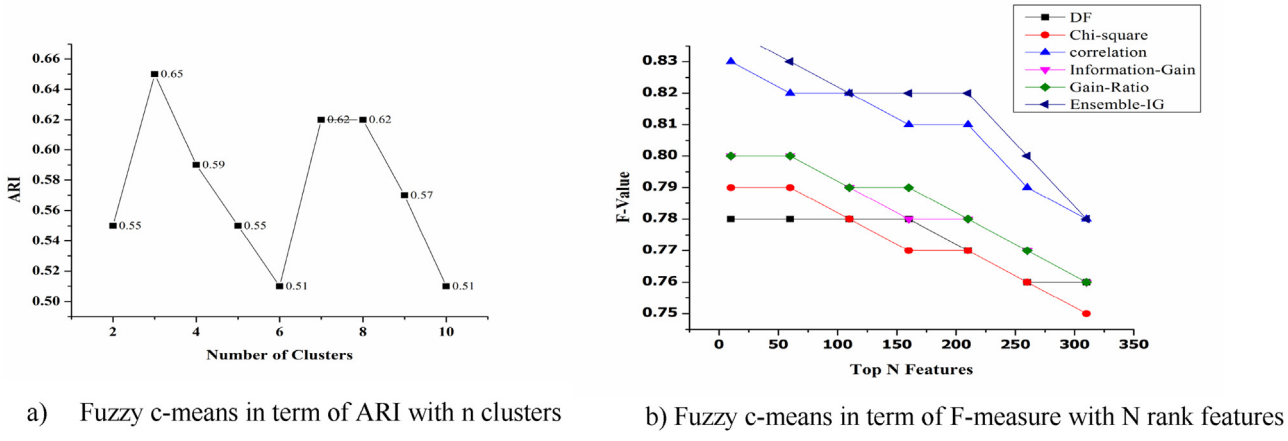
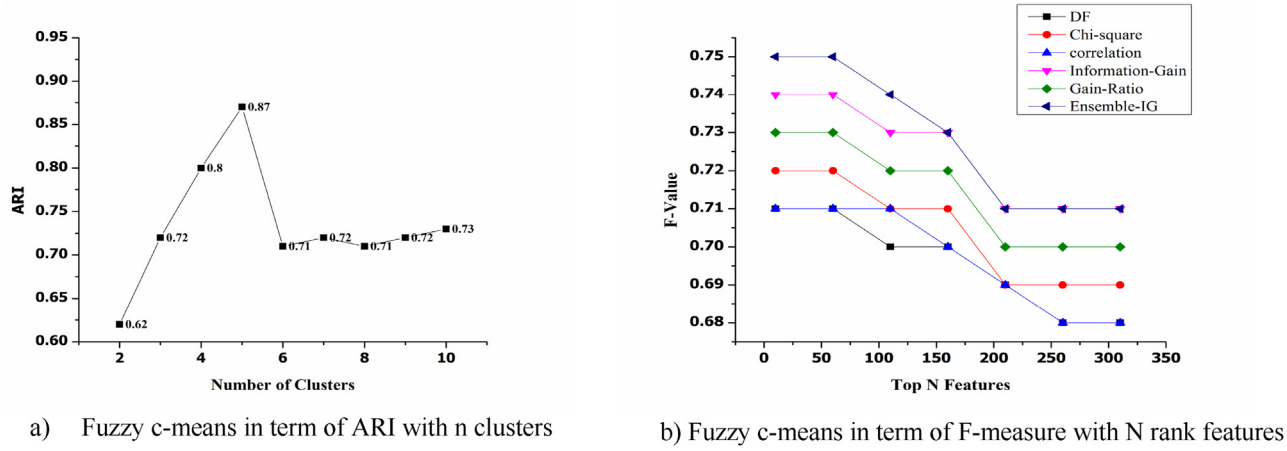


Fig. 5. Evaluation results of Fuzzy c-means on GoF patterns collection.
a) Fuzzy c-means in term of ARI with n clusters b) Fuzzy c-means in term of F-measure with N rank features

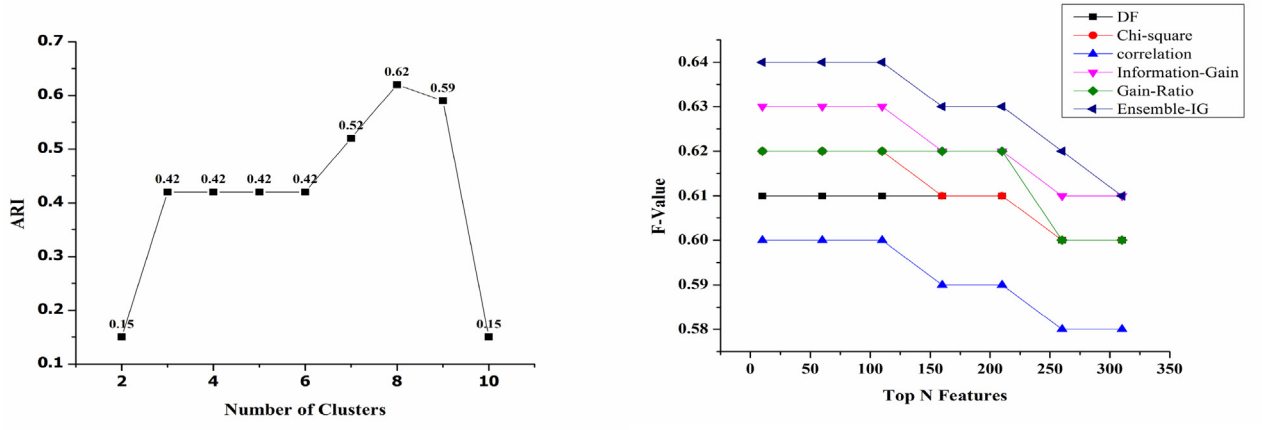
highest F-measure is suggested as a best weighting method for the Fuzzy c-means. While Fig. 6-a depict the evaluation results of Fuzzy c-means (with the best weighting method) with a different number of clusters k (each cluster is considered as pattern class). According to ARI criterion, Fuzzy c-means recommend a number of design pattern classes (k = 5 in Fig. 6-a) with highest ARI, which is correct according to expert's opinion [16]. The classification of design patterns into an appropriate number of design pattern classes suggests the applicability of the proposed approach. Subsequently, we applied six feature selection methods, including our proposed technique named Ensemble-IG for the Fuzzy c-means unsupervised learner, and evaluation results are depicted in Fig. 6-b. The x-axis of Fig. 6-b (labeled with Number of Rank Features) refers to top n features rank by each feature selection method. According to F-measure criterion, the feature selection method with the highest increase in performance of Fuzzy c-means as compared to its actual performance F-measure = 0.71 before feature selection (illustrated in Fig. 4-b) is suggested. For example, with top 10 features (depicted on the x-axis of Fig. 6-b) rank by each feature selection method, there is an increase from 1.84% to 5.05% (in term of F-measure) in the performance of Fuzzy c-means. However, the proposed feature selection method Ensemble-IG with the highest increase in performance (i.e. 5.05%) is suggested for the fuzzy c-means. We have



a) Fuzzy c-means in term of ARI with n clusters

b) Fuzzy c-means in term of F-measure with N rank features

Fig. 6. Evaluation results of Fuzzy c-means on Douglass patterns collection.
a) Fuzzy c-means in term of ARI with n clusters b) Fuzzy c-means in term of F-measure with N rank features



a) Fuzzy c-means in term of ARI with n clusters

b) Fuzzy c-means in term of F-measure with N rank features

Fig. 7. Evaluation results of Fuzzy c-means on Security patterns collection.
a) Fuzzy c-means in term of ARI with n clusters b) Fuzzy c-means in term of F-measure with N rank features

replicated this procedure with Top n (where n ranges from 50 to 350) features and illustrated the evaluation results in Fig. 6-b.

8.1.3. Security design patterns collection evaluation

The patterns in Security collection are divided into eight categories. According to F-measure criterion, the evaluation result in the selection of best weighting method for the Fuzzy c-means is shown in Fig. 4-c. For example, TFC with highest F-measure is suggested as a best weighting method for the Fuzzy c-means. While Fig. 7-a depict the evaluation results of Fuzzy c-means (with best weighting method) with a different number of clusters k (each cluster is considered as pattern class). According to ARI criterion, Fuzzy c-means recommend a number of design pattern classes (k = 8 in Fig. 7-a) with highest ARI, which is correct according to expert's opinion [36]. The classification of design patterns into an appropriate number of design pattern classes suggests the applicability of the proposed approach. Subsequently, we applied six feature selection methods, including our proposed technique named Ensemble-IG for the Fuzzy c-means unsupervised learner and evaluation results are depicted in Fig. 7-b. The x-axis of Fig. 7-b (labeled with Number of Rank Features) refers to top n features rank by each feature selection method. According to F-measure criterion, the feature selection method with the highest increase in performance of Fuzzy c-means as compared to its actual performance F-measure = 0.71 before feature selection (illustrated in Fig. 4-c) is suggested. For example, with top 10 features (depicted on the x-axis of Fig. 7-b) rank by each feature selection method, there is an increase from 5.08% to 12.28% (in term of F-measure) in the performance of Fuzzy c-means. However, the proposed feature selection method Ensemble-IG with the highest increase in performance (i.e. 12.28%) is suggested for the fuzzy c-means. We have replicated this procedure with Top n (where n ranges from 50 to 350) features and illustrated the evaluation results in Fig. 7-b.

8.2. Learning evaluation for design problems

The effectiveness of unsupervised learning technique Fuzzy c-means used in the proposed approach is evaluated in order to determine the design pattern class for a given real design problem using evaluation model. We consider design pattern collection (Section 6.3) for each real design problem (Section 6.4) and make the assessment using an evaluation model described in the Section 6.1.

8.2.1. Evaluation of real design problems with gang-of-Four (GoF) patterns collection

We used GoF design pattern collection (Section 6.3) and 30 GoF real problems (Section 6.4) to evaluate the efficacy Fuzzy c-means. The evaluation results are depicted in Fig. 8. According to average ARI criterion, the highest ARI value at k = 3 suggests the effective-

Table 4
Fuzzy c-means recommended class membership values for GoF design problems.

Design Pattern Class	Class Membership Degree Value Recommended By Fuzzy C-means		
	Design Problem 1	Design Problem 2	Design Problem 3
Creational	0.20	0.15	0.12
Structural	**0.67**	0.24	0.16
Behavioral	0.13	**0.61**	**0.72**

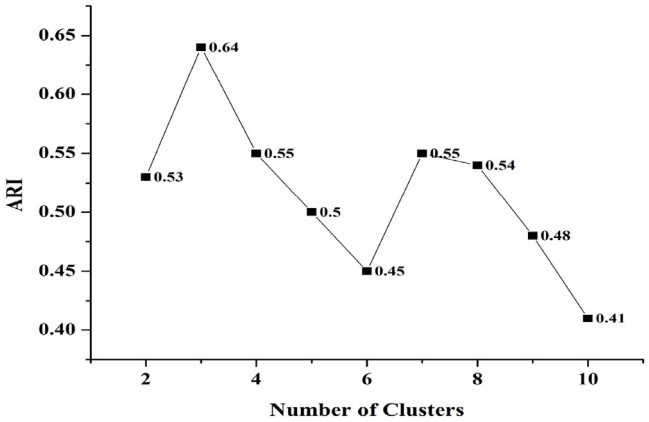


Fig. 8. Evaluation results of Fuzzy c-means in term of average ARI on GoF design patterns collection and 30 problems.

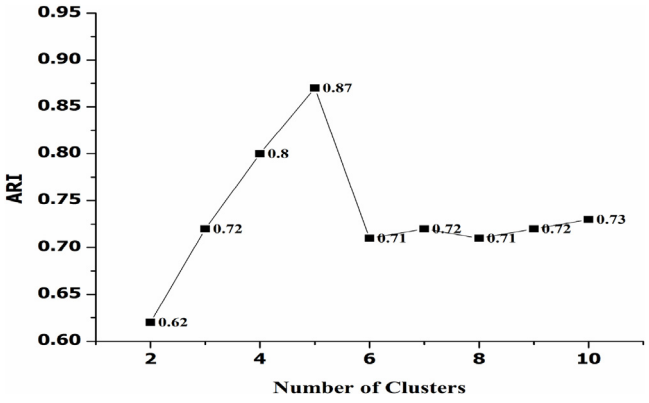


Fig. 9. Evaluation results of Fuzzy c-means in term of average ARI on Douglass design patterns collection and 20 problems.

ness of Fuzzy c-means, to classify the design problems into an appropriate number of classes. However, the closeness of design problem(s) with corresponding design pattern classes is evaluated through class membership values recommended by Fuzzy c-means. The class membership values for three GoF design problems with corresponding design pattern classes are depicted in Table 4.

The Fuzzy- c-means determine Structural, Behavioral and Behavioral design pattern class for real Design Problem 1, Design Problem 2 and Design Problem 3 respectively. Consequently, the results of the proposed approach are according to expert opinion.

8.2.2. Evaluation of real design problems with douglass patterns collection

We used Douglass design pattern collection (Section 6.3) and 20 Douglass real problems (Section 6.4) to evaluate the efficacy Fuzzy c-means. The evaluation results are depicted in Fig. 9. According to ARI criterion, the highest ARI value at k = 5 suggests the effectiveness of Fuzzy c-means used in the proposed approach, to classify the design problems into an appropriate number of classes. However, the closeness of design problem(s) with corresponding design

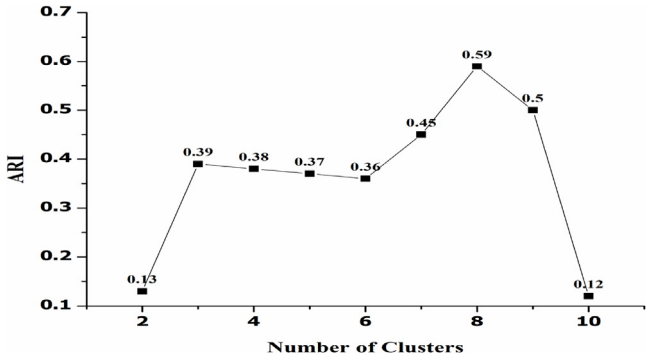


Fig. 10. Evaluation results of Fuzzy c-means in term of average ARI on Security design patterns collection and 30 problems.

pattern classes is evaluated through class membership values recommended by Fuzzy c-means. The class membership values for three Douglass design problems with corresponding design pattern classes are depicted in Table 5. The Fuzzy- c-means determine Memory, Resources and Distribution design pattern class for real Design Problem 4, Design Problem 5 and Design Problem 6 respectively. Consequently, the results of the proposed approach are according to expert opinion.

8.2.3. Evaluation of real design problems with security patterns collection

We used Security design pattern collection (Section 6.3) and 30 Security real problems (Section 6.4) to evaluate the efficacy Fuzzy c-means. The evaluation results are depicted in Fig. 10. According to ARI criterion, the highest ARI value at k = 8 suggests the effectiveness of Fuzzy c-means, to classify the design problems into an appropriate number of classes. However, the closeness of design problem(s) with corresponding design pattern classes are

evaluated through class membership values recommended by Fuzzy c-means unsupervised learner used in the proposed study. The class membership values for three Security design problems with corresponding design pattern classes are depicted in Table 6. The Fuzzy- c-means determine Access Control Models (ACM), Accounting and Enterprise Security and Risk Management (ESRM) design pattern class for real Design Problem 7, Design Problem 8 and Design Problem 9 respectively. Consequently, the results of the proposed approach are according to expert opinion.

8.3. Learning evaluation for design pattern(s) selection

In order to observe the applicability of similarity options mentioned in Section 5.4, firstly, we apply the design pattern selection method and give an analysis of nine real design problems. Subsequently, we randomly select a subset of thirty problems and present the experimental work.

8.3.1. Sample of nine design problems

The proposed approach has been applied to identify an appropriate design pattern class for each design problem mentioned in Section 6.4. In this analysis, firstly, we apply the cosine similar-

Table 5
Fuzzy c-means recommended class membership values for Douglass design problems.

Design Pattern Class	Class Membership Degree Value Recommended By Fuzzy C-means		
	Design Problem 4	Design Problem 5	Design Problem 6
Memory	**0.58**	0.09	0.07
Resource	0.10	**0.64**	0.11
Distribution	0.05	0.05	**0.62**
Concurrency	0.13	0.12	0.15
Safety and Reliability	0.14	0.10	0.05

Table 6
Fuzzy c-means recommended class membership values for Security design problems.

Design Pattern Class	Class Membership Degree Value Recommended By Fuzzy C-means		
	Design Problem 7	Design Problem 8	Design Problem 9
Accounting	0.04	**0.48**	0.05
Firewall Architecture	0.06	0.05	0.04
Access Control Models	**0.51**	0.06	0.06
Security Internet Application	0.05	0.10	0.14
Operating System Access Control	0.11	0.07	0.05
System Access Control Architecture	0.10	0.03	0.03
Identification and Authentication Patterns	0.06	0.11	0.10
Enterprise Security and Risk Management	0.07	0.10	**0.53**

Table 7
Computed ranges for $\theta 1$ and $\theta 1$ parameters.

Design Problems	Θ_1	Θ_2
Design Problem 1	0.34 to 0.50	0.0 to 0.16
Design Problem 2	0.39 to 0.65	0.0 to 0.27
Design Problem 3	0.55 to 0.71	0.0 to 0.16
Design Problem 4	0.58 to 0.71	0.0 to 0.13
Design Problem 5	0.40 to 0.63	0.0 to 0.23
Design Problem 6	0.47 to 0.68	0.0 to 0.21
Design Problem 7	0.45 to 0.67	0.0 to 0.22
Design Problem 8	0.38 to 0.72	0.0 to 0.34
Design Problem 9	0.41 to 0.64	0.0 to 0.23

ity measure (Eq. (7)) to evaluate the similarity between a design problem and patterns of the identified design pattern class. The similarity values of each design problem with patterns of its candidate design pattern class are shown in Tables A1–A9 (Appendix A) respectively. In each table, the highest similarity value between a design problem and the design pattern is highlighted. It is important to note that highlighted design patterns (in Tables A1–A9 shown in Appendix A) are computed by Eq. (7), and are identified correctly according to expert opinions. It can be concluded that for these nine design problems, in the third phase of proposed approach, when we choose Eq. (9), the highest RCD value can be obtained. However, the computation of Eqs. (10) and (11) aid to determine the ranges for the parameters $\theta 1$ and $\theta 2$ in order to select the more appropriate design pattern(s). The comparative range values for $\theta 1$ and $\theta 2$ parameters for each design problem to obtain the highest RCD is shown in Table 7.

The average RCD for all designated real design problems in term of $\theta 1$ and $\theta 2$ relevant similarity options (Section 5.4) are depicted in Fig. 11. As shown in Fig. 11(a-b), the highest RCD values 0.55 and 0.76 are obtained when $\theta 1$ and $\theta 2$ are equal to 0.4 and 0.04 respectively.

8.3.2. Sample of thirty design problems

In order to assess the accuracy of proposed approach, we randomly select 30 real design problems from the list mention in the Section 6.4. Subsequently, it is assumed that each design problem is with correctly identified (using the proposed approach) design pattern class. The average RCD for all designated real design problems in term of $\theta 1$ and $\theta 2$ relevant similarity options are depicted in Fig. 12. As shown in Fig. 12 (a-b), the highest RCD values 0.55 and 0.77 are obtained when $\theta 1$ and $\theta 2$ are equal to 0.5 and 0.03 respectively.

9. Evaluation summary

- We applied the Fuzzy c-means unsupervised learning technique in the proposed approach and achieve better results. The class membership value for a design problem is suggested by Fuzzy c-means aid to determine the design pattern class for the corresponding design problem.
- We used different weighting methods (such as TFIDF, TFC, LTC, Binary, and Entropy) in order to increase the efficacy of proposed approach, however, no single weighting method can be recommended. For example, TDIDF for GoF and Douglass pattern collections, while TFC for security design pattern collection remains more effective according to F-measure criterion.
- The results of feature selection methods for the Fuzzy c-means indicate that learning precision can be increased by reducing the features until it is found at maximum. However, we observed that the Fuzzy c-means is more sensitive to proposed feature selection method Ensemble-IG as compared to other feature selection methods. Subsequently, in our experiment, learning precision is evaluated with top N (from 10 to 350) features rank by feature selection methods. We observe an increase in the learning precision (in term of F-measure) of Fuzzy c-means with proposed Ensemble-IG as compared to others feature selection methods used in the study.
- The values of $\theta 1$ and $\theta 2$ parameters depend on the selected design pattern collection and the sample of selected design problems. The developer can replicate these experiments with any sample of design problems and can tune these parameters.

10. Conclusion and future work

The experimental results illustrate that the proposed approach based on text categorization approach via unsupervised learner Fuzzy c-means aid to provide a promising base for selection of more appropriate design pattern(s) for a particular design problem. While existing automatic techniques based on UML-based, Ontology-based, and text categorization approaches aims to reduce preprocessing cost and computation time to select a right design pattern(s) for given design problem. However, formal specification,

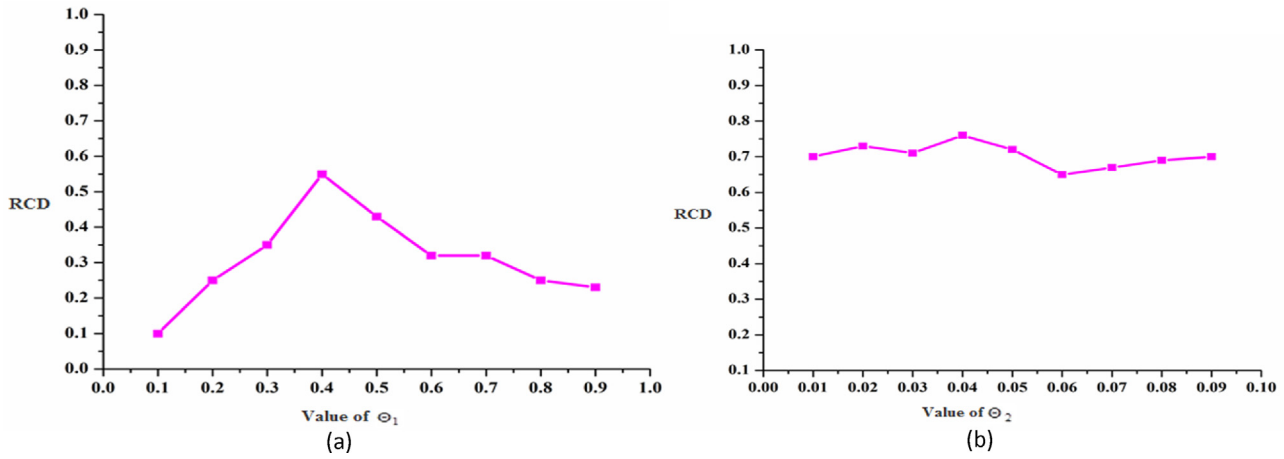


Fig. 11. Parameters of similarity options for Sample of 9 design problems.
a) Mean RCD for similarity option using Eq. (5) b) Mean RCD for similarity option using Eq. (6)

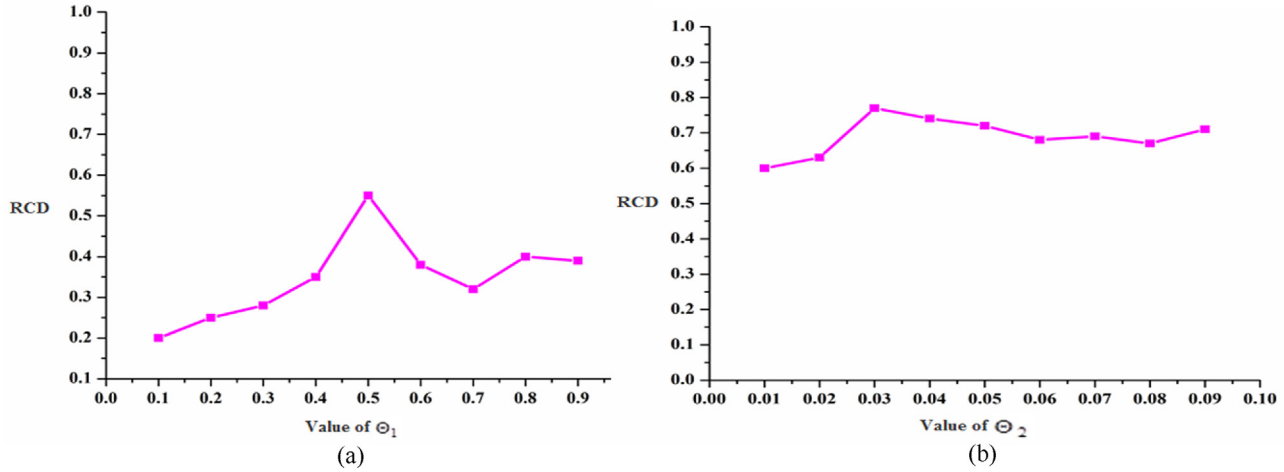


Fig. 12. Parameters of similarity options for sample of 30 design problems.
a) Mean RCD for similarity option using Eq. (5) b) Mean RCD for similarity option using Eq. (6)

the existence of the ground reality of class labels for design patterns, multi-class problem, and an adequate sample size is required for these automatic techniques to make precise learning. To address this issue, the proposed approach is evaluated with the use of three design pattern collections and descriptions of ample real design problems. Moreover, we also proposed a feature selection method Ensemble-IG by leveraging an Improved Global Feature Selection Scheme (IGFSS) and ensemble with Information Gain (IG) to overcome the multi-class problem. The proposed approach has three phases, Preprocessing, Pattern Organization and Determination of design pattern class, and Suggestion of Design Pattern(s). The Fuzzy c-means unsupervised learning technique is used in the proposed approach. We also proposed an evaluation model to evaluate the effectiveness of the proposed method.

We conclude several important consequences with respect to the experimental results. First, no single weighting method can be recommended for the Fuzzy c-means and varies across the design pattern collections. Second, in order to determine appropriate design pattern classes for given design problem(s), Fuzzy c-means achieve better results with respect to expert's opinions. Third, the effectiveness of the proposed approach is highly affected by feature selection methods, and we observe a significant increase in the precision learning of Fuzzy c-means with Ensemble-IG feature selection method. For example, we observed an increase from up to 10.52% (in term of F-measure) in the performance of Fuzzy c-means when features are selected via Ensemble-IG feature selection method. Finally, the values of $\theta 1$ and $\theta 2$ parameters depend on the selected design pattern collection and the sample of selected design problems.

In a future work, we will consider more unsupervised learner and will use the semantic of two contiguous words instead of the single word in the feature vector in order to improve the effectiveness of the proposed approach. Subsequently, we will also consider the Stack Exchange sites such as Information Security, Graphic Design, Web Application, Board and Card Games and Game Development to analyze the variation of problems and suggestion of design patterns.

Acknoweldgement

This research is supported by the City University of Hong Kong research funds (Project No. 7004683, 7004474 and 7200354).

Appendix A. : Cosine Similarity (CS) measures values for 9 design problems (Section 6.4)

Table A1
Cosine similarity results for Design Problem 1.

Structural Design Patterns	Cosine Similarity (CS)
Adapter	0.33
Bridge	**0.51**
Composite	0.31
Decorator	0.20
Fascade	0.34
Flyweight	0.20
Proxy	0.26

Table A2
Cosine similarity results for Design Problem 2.

Behavioral Design Patterns	Cosine Similarity (CS)
Chain of Responsibility	0.37
Command	0.55
Interpreter	0.30
Iterator	0.38
Mediator	**0.66**
Memento	0.39
Observer	0.37
State	0.37
Strategy	0.36
Template	0.10
Visitor	0.37

Table A3
Cosine similarity results for Design Problem 3.

Behavioral Design Patterns	Cosine Similarity (CS)
Chain of Responsibility	0.35
Command	0.18
Interpreter	0.23
Iterator	0.26
Mediator	0.35
Memento	0.38
Observer	0.55
State	0.34
Strategy	0.55
Template	0.42
Visitor	**0.72**

Table A4
Cosine similarity results for Design Problem 4.

Memory Design Patterns	Cosine Similarity (CS)
Fixed SizeBuffer	0.58
Garbage Collection	0.38
Garbage Compactor	0.55
Pool Allocation	0.28
Smart Pointer	0.57
Static Allocation	**0.72**

Table A5
Cosine similarity results for Design Problem 5.

Resource Design Patterns	Cosine Similarity (CS)
Order locking	0.18
Priority ceiling	0.40
Priority inheritance	0.24
Simultaneous locking	0.23
Critical Section	**0.64**
Highest locker	0.23

Table A6
Cosine similarity results for Design Problem 6.

Distribution Design Patterns	Cosine Similarity (CS)
Broker	0.21
Data Bus	0.47
observer	0.26
proxy	0.32
Remote method call	0.16
Shared memory	**0.69**

Table A7
Cosine similarity results for Design Problem 7.

Access Control Models (ACM) Design Patterns	Cosine Similarity (CS)
Authorization	0.40
Multilevel Security	0.24
Reference Monitor	0.34
Role Rights Definition	0.45
RoleBased Access Control	**0.68**

Table A8
Cosine similarity results for Design Problem 8.

Accounting Design Patterns	Cosine Similarity (CS)
Audit Requirements	0.25
Audit Trails and Logging Requirement	**0.73**
Intrusion Detection Requirements	0.38
Non-Repudiation Requirements	0.31
Security Accounting Requirements	0.34

Table A9
Cosine similarity results for Design Problem 9.

Enterprise Security and Risk Management (ESRM) Design Patterns	Cosine Similarity (CS)
Enterprise Partner Communication	0.36
Enterprise Security Approaches	0.39
Enterprise Security Services	0.41
Risk Determination	0.33
Security needs identification for enterprise asset	0.25
Threat Assessment	**0.65**
Vulnerability Assessment	0.21

Appendix B. : Ensemble-IG (Proopsed feature selection method)

The Ensemble-IG feature selection method is proposed by leveraging an improved global feature selection scheme in order to overcome the issue of the multi-class problem (in the text categorization) as exist in the classic form of Information Gain (IG). For example, in the classic form of IG, single score of each feature f is calculated using the Eq. (A1).

$$\mathrm{IG(f)} = -\sum_{k=1}^{NC} P(C_k)\log P(C_k) + P(f)\sum_{k=1}^{NC} P(C_k|f)\log P(C_k|f) + P(\bar{f})\sum_{k=1}^{NC} P(C_k|\bar{f})\log P(C_k|\bar{f}) \quad \text{(A1)}$$

In the Eq. (A1), the NC, $P(C_k)$, $P(f)$, $P(\bar{f})$, $P(C_k|f)$, and $P(C_k|\bar{f})$ present the number of classes, probability of class C_k, probability for the presence of the feature f, probability for the absence of the feature f, probability of class C_k for the presence of feature f, and the probability of class C_k for the absence of feature f respectively. The workflow of the proposed Ensemble-IG is based on the improved global feature selection scheme by ensemble the classic IG and one sided OR (Odd Ratio) feature selection method, as follows. **Procedure Ensemble-IG (VSM, AL, CL, FL, SFL, FFL, n)**

Input:

VSM = Vector Space Model

AL = {Set of Actual Labels}, For example, in the case of GoF design pattern collection, Structural, Creational and Behavioral class labels are used.

CL = {Set of Created Labels with $NC * 2$}, where NC is the number of classes (or cardinality of AL)

FL = Actual Feature List

SFL = Sorted Feature List

FFL = Final Feature List
n = The value of n referred to top n features
Begin
Step-1
The first step is used to assign labels to features.

- Compute the one-sided local feature selection score (using Odd Ratio) of each feature of FL for each class of AL.
- The new label set CL is created with NC^*2labels to describe the membership and non-membership to the classes of AL.
- For each feature of FL, compute the local feature selection score, and assign the labels according to highest score.

Step-2
The second step is to apply the classic IG.

- Compute the IG feature selection score for each feature.
- Create a new sorted (in descending order) feature list SFL

Step-3
The third step is to construct a new feature set.

- Traverse each feature of SFL and create a FFS with a size equal to the cardinality of FL and set of negative feature ratio in the range from 0 to 1.
- The constructed FFS should be representative of each class of AL

Step-4
The fourth step described the end of the Ensemble-IG workflow.

- If the cardinality of FFL is less than the from the cardinality of FL, then add the disregarded features with highest global feature selection score to FFL.

End
Output:
The creation of new VSM which contain the equal number of features for the each class of AL.

Appendix C. : Examples to describe the implementation of the proposed Approach

Example 1: Implementation of the proposed approach for the organization of GoF design patterns

Step 1
The removal of the stopwords, words stemming, and indexing activities are performed to create the vector space model for the N design patterns (i.e. N = 23) and M unique words (i.e. M = 1465). Though we retrieved M = 1465 features (Step-1), however, due to space limitation, we are describing the vector space model with M = 10 as follows (Table C1).
Step 2
We applied the weighting methods (Section 5.1). However, in Table B1, we present the VSM with the binary weighting method.
Step 3
In Table C2, we described the analysis result of Fuzzy c-means. The performance of fuzzy c-means with different weighting methods is assessed using the evaluation criteria described in Section 6.1. According to the evaluation criteria, the TFIDF (with F-measure = 0.73) is the best weighting method for the Fuzzy c-means in the case of the organization of the GoF design patterns.
Step 4
Though we evaluated the effectivenessand robustness of the proposed approach with the different number of clusters (Fig. 5-a), however, with 3 clusters (i.e. k = 3, according to expert's opinion), in terms of ARI, the performance of Fuzzy c-means (ARI = 0.65) remain better than as compared to other values of k.

Example 2: Implementation of the proposed approach for determination of a pattern class for the Design Problem 1 (DP-1) in the context of GoF design patterns

The description of design problem 1 (DP-1) [4] described in the context of GoF pattern collection is as follows. The keywords which are used in Table A9 are highlighted in the description of DP-1.
Design Problem 1: "Design a system enabling to display on a screen some empty windows (no button, no menu. . .). A window can have several **differ**ent styles depending on the platform used. We consider two platforms, XWindow and Presentation Manager. The **client** code must be written independently and without knowledge of the future **execution** platform. It is probable that the system evolves in order to display specialized windows by '**application** windows' (able to manage applications) and 'iconised windows' (with an icon)".
Though the Structural pattern class has been mapped for the design problem 1(DP-1) [4], however, in order to determine the appropriate pattern class for DP-1 using the proposed approach, the following steps are used.
Step 1. The removal of the stopwords, words stemming and indexing activities are performed to create the vector space model for the N design patterns (i.e. N = 23), one design problem (Design Problem 1 (DP-1), and M unique words (i.e. M = 1471). Though we retrieved M = 1471 features, however, due to space limitation, we are describing the vector space model with M = 10 and binary weighting method, shown in Table C3.
Step 2. We applied the weighting methods (Section 5.1). However, in Table C3, we present the VSM with the binary weighting method.
Step 3. The Fuzzy c-means with its best weighting method (i.e. TFIDF) is assessed to determine the appropriate pattern class for the DP-1. The results are shown in Table C4. The results of Table C4 depict the effectiveness of the proposed approach (via Fuzzy c-means) to determine the right pattern class (i.e. Structural according to expert opinion [4]) for the DP-1. The list of patterns belongs to the suggested pattern class (which is determined via Fuzzy c-means) for the design problem-1 (DP-1), is shown in Table A1.
Step 4. The Step 1 to Step 3 can be repeated for the rest of the problems [4,18,38,39] related to target design pattern collections [3,16,36].

Table C1
Vector Space Model for GoF design pattern collection.

N Patterns	Client	Class	Application	Extend	Hierarchy	System	Structure	Behavior	Differ	Execution
1	0	0	0	1	1	1	0	0	1	0
2	1	0	0	0	0	0	0	0	1	0
-	-									
23	0	0	1	0	0	1	0	0	1	1

Table C2
Performance assessment of Fuzzy c-means with weighting methods.

GoF Design Patterns			Fuzzy c-means with Weighting Methods				
Pattern	Category (By Expert)	Cluster Number	Binary	Entropy	LTC	TFC	TFIDF
Abstract Factory	Creational	1	1	2	1	1	2
Adapter	Structural	2	2	2	3	2	2
Bridge	Structural	2	2	2	3	2	2
Builder	Creational	1	1	1	1	3	1
Chain of Responsibility	Behavioral	3	3	1	2	3	1
Command	Behavioral	3	3	3	2	3	1
Composit	Structural	2	2	1	1	3	1
Decorator	Structural	2	2	2	2	3	1
Factor Methods	Creational	1	1	1	1	2	2
Fascade	Structural	2	2	1	1	2	2
Flywieght	Structural	2	2	2	2	3	1
Interpreter	Behavioral	3	3	3	1	2	2
Iterator	Behavioral	3	3	2	2	3	1
Mediator	Behavioral	3	3	3	2	3	1
Memento	Behavioral	3	3	3	3	3	1
Observer	Behavioral	3	3	1	1	3	1
Prototype	Creational	1	1	1	2	3	2
Proxy	Structural	2	2	1	2	3	1
Singleton	Creational	1	1	1	1	1	2
State	Behavioral	3	3	1	3	3	1
Strategy	Behavioral	3	3	3	1	2	3
Template	Behavioral	3	3	1	1	2	2
Visitor	Behavioral	3	3	3	3	2	3

Table C3
Vector Space Model for GoF design patterns and Design Problem 1(DP-1).

N + 1 Patterns +Problem	Client	Class	Application	Extend	Hierarchy	System	Structure	Behavior	Differ	Execution
1	0	0	0	1	1	1	0	0	1	0
2	1	0	0	0	0	0	0	0	1	0
– –										
23	0	0	1	0	0	1	0	0	1	1
24	1	0	1	0	0	1	0	0	1	1

Table C4
Performance assessment of Fuzzy c-means (with best weighting method TFIDF) to determine a pattern class for the Design Problem (DP-1).

Pattern	Category (By Expert)	Cluster Number	c-means + TFIDF
Abstract Factory	Creational	1	1
Adapter	Structural	2	2
Bridge	Structural	2	2
Builder	Creational	1	1
Chain of Responsibility	Behavioral	3	1
Command	Behavioral	3	1
Composit	Structural	2	2
Decorator	Structural	2	2
Factor Methods	Creational	1	1
Fascade	Structural	2	2
Flywieght	Structural	2	2
Interpreter	Behavioral	3	3
Iterator	Behavioral	3	1
Mediator	Behavioral	3	3
Memento	Behavioral	3	1
Observer	Behavioral	3	1
Prototype	Creational	1	3
Proxy	Structural	2	2
Singleton	Creational	1	3
State	Behavioral	3	1
Strategy	Behavioral	3	3
Template	Behavioral	3	1
Visitor	Behavioral	3	3
Design Problem-1	Structural	2	2

a) Example 3: Implementation of proposed approach for the selection of right design pattern(s)

The right pattern class and the list of correlated design patterns for Design Problem 1 (DP-1) is determined through the unsupervised learner (Fuzzy c-means) of the proposed approach is shown in Table A1. The main steps to select the right design pattern(s) for the Design Problem 1 (DP-1) as follows.

Step 1. The cosine similarity between the feature vectors of DP-1 and patterns of the corresponding pattern class (determine through proposed approach) is computed using the Eq. (7). The results are shown in Table A1 (Appendix A)

Step 2. We applied the EqS. (9)–(11) to suggest the right design patter(s) for the DP-1. The thresholds Θ_1 and $\Theta_2$are suggested on the base of cosine values retrieved via the Eq. (7).

a) The first similarity option (i.e Eq. (9)) aid to recommend the right design pattern (i.e. The Bridge pattern with the highest cosine value 0.51) for the DP-1.
b) In the case of second similarity option, the suggested threshold value Θ_1 is 0.33–0.51 (Table 7) to recommend the zero, one or more than one design pattern(s).
c) In the case of third similarity option, the suggested threshold value Θ_2 is 0.0–0.16 (Table 7) to recommend design patterns contiguous to the best design pattern (with CSmax)

Step 3. The same procedure is repeated for the rest of design problems.

Step 4. Finally, the Eq. (17) is used to select the best similarity option (Section 5.4) by considering the ratio of the number of right suggested and total suggested design patterns. In the case of Design Problem 1(DP-1), the first similarity option is suggested to select the right design pattern.

References

[1] L. Bass, P. Clements, R. Kazman, Software Architecture in Practice, 3rd ed., Addison-Wesley Professional, 2012.
[2] W.G. Wood, A Practical Example of Applying Attribute-Driven Design (ADD), Version 2.0, Technical Report Software Engineering Institute, 2007.
[3] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, Boston, MA, 1995.
[4] C. Bouhours, H. Leblance, C. Percebois, Spoiled patterns: how to extend the GoF, Softw. Q. J. 23 (2015) 661–694.
[5] G. Booch, Handbook of Software Architecture, http://handbookofsoftwarearchitecture.com/.
[6] H. Baraki, et al., Interdisciplinary design patterns for socially aware computing, Procedding 37th International Conference on Software Engineering (ICSE) (2015).
[7] D.K. Kim, C.E. Khawand, An approach to precisely specifying the problem domain of design patterns, J. Visual Lang. Comput. 18 (2007) 560–591.
[8] N.L. Hsueh, J.-Y. Kuo, C.-C. Lin, Object-Oriented design: a goal-driven and pattern-based approach, J. Softw. Syst. Model. 8 (1) (2007) 1–18.
[9] D.K. Kim, W. Shen, Evaluating pattern conformance of UML models: a divide and conquer approach and case studies, Softw. Q. J. 16 (3) (2008) 329–359.
[10] S. Hasso, C.R. Carlson, A theoretically-based process for organizing design patterns, Proceedings of 12th Pattern Language of Patterns (2005).
[11] E. Blomqvist, Pattern ranking for semiautomatic ontology construction, Proceedings of SAC (2008).
[12] P.E. Khoury, A. Mokhtari, E. Coquery, M.S. Hacid, An ontological interface for software developers to select security patterns, Proceedings of 19th International Conference on Database and Expert Systems Application, (DEXA'08) (2008) 297–301.
[13] S.M.H. Hasheminejad, S. Jalili, Design patterns selection: an automatic two-phase method, J. Syst. Softw. 85 (2012) 408–424.
[14] A. Hotho, A. Nurnberger, G. Paab, A brief survey of text mining, J. Comput. Linguist. Lang. Technol. 20 (2005) 19–62.
[15] A.K. Uysal, An improved global feature selection scheme for text classification, Expert Syst. Appl. 43 (2016) 82–92.
[16] B.P. Douglass, Real-Time Design Patterns: Robust Scalable Architecture for Real-Time Systems, Addison-Wesley/Longman Publishing Co., Inc., Boston, MA, USA, 2002.
[17] L. Rising, The Pattern Almanac 2000, Addison-Wesley, 2000.
[18] A. Silberschatz, P.B. Galvin, G. Gagne, Operating System Concepts, 6 ed., 2002.
[19] W. Pree, Design Patterns for Object-Oriented Software Development, ACM Press/Addison-Wesley, 1995.
[20] P. Coad, D. North, M. Mayfield, Mayfield, Object Models: Strategies, Patterns, Applications, Yourdon Press, NJ, USA, 1995.
[21] W.F. Tichy, A catalogue of general-Purpose software design patterns, Proceedings of Technology of Object-Oriented Languages and Systems (1997) 330–339.
[22] W. Zimmer, Relationships between design patterns, Journal of Pattern Languages of Program Design 1 (1995) 345–364.
[23] G.J. Kim, J.S. Han, Clustering algorithm of design pattern using object-Oriented relationship, Proceeding of Computational Science and Its Applications LNCS, Springer, ICCSA (2007) 997–1006.
[24] P. Velasco-Elizondo, R. Marin-Pina, S. Vazquez-Reyes, A. Mora-Soto, J. Mejia, Knowledge representation and information extraction for analyzing architectural patterns, science of computer programming, Sci. Comput. Programm. 121 (2016) 176–189.
[25] Q. Wu, Ye. Zhang, K. Ng, S.-S. Ho, FORESTEXTER: An efficient random forest algorithm for imbalanced text categorization, Knowl. Based Syst. 67 (2014) 105–116.
[26] M.F. Porter, An algorithm for suffix stripping, journal of, Program Electron. Libr. Inf. Syst. 40 (2006) 211–218 (p).
[27] P.D. Turney, P. Pantel, From frequency to meaning: vector space models of semantics, J. Artif. Intell. Res. 37 (2010) 141–188.
[28] K. Frantzi, S. Ananiadou, H. Mima, Automatic recognition of multi-word terms:. the c-value/nc-value method, Int. J. Digital Libr. 3 (2) (2000) 115–130.
[29] I. Idris, A. Selamat, Improved email spam detection model with negative selection algorithm and particles warm optimization, Appl. Soft Comput. 22 (2014) 11–27.
[30] E. Sarac, S.A. Ozel, An ant colony optimization based feature selection for web page classification, Sci. World J. (2014) 1–16.
[31] W. Medhat, A. Hassan, H. Korashy, Sentiment analysis algorithms and applications: a survey, Shams Eng. J. 5 (2014) 1093–1113.
[32] C. Zhang, X. Wu, Z. Niu, W. Ding, Authorship identification from unstructured texts, Knowl. Based Syst. 66 (2014) 99–111.
[33] M. Ricardo, et al., PreText: A Simple Text Preprocessing Tool, 2017 http://sites.labic.icmc.usp.br/torch/msd2011/jpretext/.
[34] E. Alpaydın, Introduction to Machine Learning, 2nd ed., The MIT Press, Cambridge, Massachusetts, London, England, 2010.
[35] J.C. Bezdek, R. Ehrkich, W. Full, The fuzzy c-Means clustering algorithm, J. Comput. Geosci. 3 (1984) 191–203.
[36] J. Wu, H. Xiong, C. Liu, J. Chen, Generalization of distance functions for fuzzy c-Means clustering with centroids of arithmetic means, IEEE Trans. Fuzzy Syst. 20 (3) (2012).
[37] C.C. Aggarwal, A. Hinneburg, D.A. Keim, On the surprising behavior of distance metrics in high dimensional spaces, Proceeding of the 8th International Conference on Database Theory (2001) 420–434.
[38] S. Miyamoto, K. Honda, H. Ichihashi, Algorithms for Fuzzy Clustering: Methods in c-means Clustering with Application, Studies in Fuzziness and Soft Computing, Springer, 2008, pp. 229.
[39] L. Ertoz, M. Steinbach, V. Kumar, Finding clusters of different sizes shapes and densities in noisy, high dimensional data, Proceeding of the Second SIAM International Conference on Data Mining (2003).
[40] K.L. Du, M.N.S. Swamy, Fundamentals of machine learning, in: Neural Networks and Statistical Learning, 2014.
[41] A. Huang, Similarity measures for text document clustering, Proceedings of NZCSRSC (2008).
[42] F. Sebastiani, Machine learning in automated text categorization, ACM Comput. Surv. 34 (1) (2002) 1–47.
[43] N.X. Vinh, J. Epps, J. Bailey, Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance, J. Mach. Learn. Res. 11 (2010) 2837–2854.
[44] P. Rousseeuw, A. Silhouettes:, Graphical aid to the interpretation and validation of cluster analysis, J. Comput. Appl. Math. 20 (1) (1987) 53–65.
[45] M. Schumacher, E. Fernandez, D. Hybertson, D. Buschmann, F. Security Patterns: Integrating Security and Systems Engineering, John Wiley and Sons, 2006.
[46] G. Forman, An extensive empirical study of feature selection metrics for text classification, J. Mach. Learn. Res. 3 (2003) 1289–1305.
[47] J.M. Santos, M. Embrechts, On the use of the adjusted rand index as a metric for evaluating supervised classification, Proceeding of the 19th International Conference on Artificial Neural Netwrokds (2009) 175–184.
[48] G. Saporta, G. Youness, Comparing Two Partitions: Some Proposals and Experiments, Chapter of COMPSTAT, Springer, 2002, pp. 243–248.
[49] A.S. Tanenbaum, H. Bos, Modern Operating Systems, Fourth Edition, Pearson Education Limited, 2015.
[50] A. Shalloway, R. Trott, Design Pattern Explained: A New Perspective on Object Oriented Design, Addison Wesley, 2001.
[51] A. Shvets, Design Pattern Explained Simply, 2010, pp. 117 https://sourcemaking.com/design-patterns-ebook.
[52] T. Hastie, R. Tibshirani, Friedman, The Element of Statistical Learning: Data Mining, Inference and Prediction, Springer Series in Statistics, 2nd ed., 2008, pp. 764.