Expert Systems With Applications 154 (2020) 113441

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Swarm optimized cluster based framework for information retrieval

Amol P. Bhopale[*], Ashish Tiwari

Department of Computer Science and Engineering, Visvesvaraya National Institute of Technology, Nagpur, India

A R T I C L E   I N F O

Article history:
Received 15 April 2019
Revised 31 March 2020
Accepted 6 April 2020
Available online 11 April 2020

Keywords:
Information retrieval
Swarm intelligence
Big data clustering
Frequent pattern mining
Unsupervised learning

A B S T R A C T

This work explores the integrated power of swarm intelligence and advances in data mining techniques to solve the information retrieval (IR) problem of rapidly growing digital content on the World Wide Web. We propose a swarm optimized cluster based framework with frequent pattern mining techniques to retrieve user-specific knowledge from extensive document collections. In the pre-processing phase, we split the task into two sub-tasks. The first is to decompose the document collection into groups using a bio-inspired K-Flock clustering algorithm, while the second extracts frequent patterns from each cluster using a memory-efficient Recursive Elimination (RElim) algorithm. In the next phase, we implement a cosine similarity based probabilistic model to retrieve query-specific documents from clusters based on the matching scores between the closed frequent patterns of queries and clusters. The performance of a system is evaluated by conducting several experiments which are carried out on five well-known, diverse and variable size datasets viz- TREC 2014-15 CDS (Clinical Decision Support) datasets containing 733,138 records, OHSUMED dataset with 348,566 records from Medline database, NPL dataset with 11,429 records, LISA document collection of 6004 records, CACM (Collection of ACM) dataset of 3204 records. The results show that the proposed IR framework significantly outperforms the traditional sequential IR approach and other state-of-the-art IR approaches, both in terms of the quality of the returned documents and the time of execution.

1. Introduction

In the era of digitization, the rapid adoption of massive digital content available online has made it more complex and cumbersome to retrieve user specific content from a large corpus. Information retrieval techniques have a wide range of applications in the fields of research publications, academics, e-commerce, clinical decision support, and in many other similar areas where retrieval time, as well as accuracy, is considered. IR is the process of digging out query specific text, image, or multimedia information from the web content. The retrieval of information in horizontal searches, i.e., searches in general, yields much less precision compared to topical or vertical searches. Researchers are, therefore, constantly working on information retrieval techniques to improve precision. Salton & McGill, 1983 have defined information retrieval as a process where $Q = \{t_1, t_2, \dots, t_m\}$ is the user defined query for which relevant documents or information is required to be extracted from the $D = \{d_1, d_2, d_3, \dots, d_n\}$ collection of n documents based on matching scores, where each document consists of $d_1 = \{t_1, t_2, t_3, \dots, t_k\}$ k terms. This process takes longer execution time, i.e., shows polynomial time complexity, so dealing with a large amount of data is highly inappropriate.

Over the last few years, an immense amount of research has been undertaken in the development of IR techniques. Due to overwhelming information sources, it is becoming more and more challenging to understand the context and to get the relevant results for user queries from the dataset. As a result, the topical search came into the picture that used data categorization for searching. But what if the data is unknown and the context is not easily understood, in such cases unsupervised technique such as clustering is used to decompose large datasets into small groups for faster and more accurate access to information.

Traditional partitioned based K-means clustering is a benchmark algorithm for document clustering. It has been widely used in a number of applications due to its simplicity and computational efficiency. However, the efficiency of the algorithm is very sensitive to the initial selection of cluster centroids. With these random centroid values, each algorithm produces a variety of solutions and can, therefore, plunge into local optima. Thus, the final quality of clusters falls short from the global best. Although it is possible to test all possible partitions for globally optimal results through a brute-force approach, it is not feasible for data-points in large datasets. The performance of K-means clustering algorithm is also susceptible to data distribution; it generally produces good results

[*] Corresponding author.
E-mail addresses: amolpbhopale@gmail.com (A.P. Bhopale), at@cse.vnit.ac.in (A. Tiwari).

https://doi.org/10.1016/j.eswa.2020.113441
0957-4174/© 2020 Elsevier Ltd. All rights reserved.

with evenly distributed data. Therefore, in order to address these limitations, different nature-inspired heuristic techniques are used for clustering problems. Swarm intelligence based optimization algorithms with multiple search agents have the advantage of guiding iterative computing to search for global optima while avoiding local optima. Mimic the swarming behavior allows these agents to cooperatively steer towards an optimal objective within a reasonable time. Therefore, in this paper, we studied the bio-inspired flock-based algorithm proposed by Cui, Gao, and Potok (2006) for document clustering.

We modified the underlying algorithm architecture to generate K fuzzy clusters. The proposed K-Flock algorithm is used in the IR framework to facilitate the retrieval process and improve the quality of results. Each document is treated as a boid, i.e., bird-like object, and the movement of these boids are governed by four steering rules viz- cohesion, separation, alignment, and Similarity-Dissimilarity. In each iteration, boids are steered towards a global optimum goal and, thus, does not fall into local optima. For all boids, fitness values w.r.t cluster centroids are calculated using Schaffer's F6 function and allowed to carry membership of more than one cluster, therefore form fuzzy clusters.

We have extended the swarm optimized document decomposition approach with advances in data mining techniques for the IR task. From the document clusters, we extracted closed frequent patterns using memory efficient Recursive Elimination (RElim) algorithm. These closed patterns are further used to assign probability values to each cluster, based on which the relevant documents are retrieved. The main contributions of this paper are as follows:

- We have proposed an unsupervised, bio-inspired K-Flock algorithm for document decomposition that forms clusters over a large set of document collections by considering the collective behavior of document objects in virtual space. The beauty of the algorithm lies in the movement of document objects that follow simple rules, exactly same as some biological objects. The algorithm also allows document objects to have more than one cluster membership to improve relevancy results. The performance of clusters are evaluated using the generalized intra-inter silhouette index proposed in Rawashdeh & Ralescu, 2012.
- We have employed memory efficient Recursive Elimination (RElim) algorithm for extracting closed frequent patterns (CFPs) i.e., concepts from clusters. This algorithm generates number of topics from each cluster.
- We have proposed the Probabilistic Document Retrieval Model (PDRM) for query specific document retrieval. CFP's extracted using the RElim algorithm are used in PDRM to assign probabilities based on matching score with query key-phrases. These probability values are used to rank clusters for document extraction specific to each query.
- Extensive experimentations are performed on diverse and variable size datasets to examine the proposed approach. Significant improvements have been observed over the state-of-the-art IR approaches.

These contributions help to find out solutions of the following research questions-

(Q1) How to model the flocking behavior of biological objects for document clustering, and use their swarm intelligence to achieve the globally optimized solution?

(Q2) Does the memory efficient RElim frequent pattern mining algorithm extract countable informative patterns from the large, noisy datasets?

(Q3) Will the proposed framework give a new direction for research on integrating bio-inspired techniques with advances in data mining for information retrieval?

Rest of the paper is arranged as follows: In Section 2, we discuss literature work performed in the information retrieval field along with the swarm intelligence approaches. Section 3 presents the proposed framework and K-Flock, RElim, and PDRM algorithms followed by experimental details, performance evaluation measure, parameter settings, and results in Section 4. In Section 5, we conclude the work with remarks and future scope.

2. Related work

The massive increase in the amount of digital content on the World Wide Web causes critical problems in generating precise solutions for information retrieval tasks. However, horizontal retrieval of information, i.e., across all domains, is complicated when it comes to a large volume of information resources. IR tasks have been studied by several researchers (Blei, Ng, & Jordan, 2003; Croft & Harper, 1979; Lafferty & Zhai, 2001; Ponte & Croft, 1998; Wei & Croft, 2006) in the past, and most of the studies discussed traditional data mining techniques, which are time-consuming on a large corpus. To retrieve query relevant information from such substantial volume resources in a reasonable time, we present K-Flock, a bio-inspired swarm intelligence model combined with frequent itemset mining. Aim of using swarm intelligence along with closed frequent mining techniques is to provide a more approximate solution for the IR problem in sensible time complexity.

We surveyed information retrieval literature papers and briefly classified work in three categories viz.: Pattern Mining approaches, Clustering approaches, and Bio-inspired approaches.

2.1. Pattern mining approaches

Information retrieval task has a long history of experimenting with various techniques for better retrieval results, and is still underway. In the early years, the matching keyword query and vector space model for document representation were used to generate matching scores as a measure of relevance (Wiemer-Hastings, Wiemer-Hastings, & Graesser, 2004). Various changes in VSM have been observed, such as the term frequency inverse document frequency (Tfidf) model (Salton & McGill, 1983), the Okapi BM25 model (Robertson et al., 1995), based on different term weighting selection criteria. However, VSMs are unable to deal with the semantic relationship between words and documents. Thus, in order to overcome this limitation, latent semantic indexing (LSI) and probabilistic language models got evolved in later years. The paper (Kontostathis, 2007) discussed various aspects of the LSI technique and developed a model for the retention of dimensions containing term relationship, i.e., latent semantic information. The study confirms that the optimized LSI system captures term relationship information only within the first few dimensions referred to as Essential dimensions. Recent studies have suggested some variations in semantic representation based models such as Distribution Semantic based query expansion (da Silva & Maia, 2019), self-adapting Saturated Density Function (SDF) probabilistic IR model (Song, Hu, He, & Dou, 2019).

Probabilistic language models are designed to characterize terms in the probability distribution of the relevant topical content. The language model calculates the term weights by considering their distribution over the collection of documents. It determines the relevance of the D document to the Q query. Various neural IR-based word representation studies (Marchesin, Purpura, & Silvello, 2019; Ran, He, Hui, Xu, & Sun, 2017; Yang, He, Li, & Xu, 2017) are conducted over the years to develop semantic representation words by dense vectors of real-valued numbers computed using the surrounding context. These models mostly focused on establishing semantic relationships between the words and improving the relevance of document retrieval.

Several researchers have discussed data mining approaches and improved in the IR process. Beil, Ester, & Xu, 2002 proposed the association rule mining approach called HFTC (Hierarchical Frequent Term-based Clustering). Authors used apriori algorithm to extract closed frequent itemsets and later clustered documents where these frequent patterns occurred. Fung, Wang, & Ester, 2003 introduced frequent itemset based hierarchical clustering approach to represent document collection as a hierarchical tree. A similar approach is studied in Yu, Searsmith, Li, & Han, 2004 and presented a topic modeling based clustering for Information Retrieval. Topics are discovered using frequent item-set mining algorithms. Hierarchical relationships are established between k^{th} and $k-1^{th}$ itemset, which improved precision, but this approach fails to deal with overlapping clusters for highly correlated documents. In order to reduce the noise between the user's request and the patterns available in the document set, Zhong, Li, and Wu (2012) have proposed a Pattern Taxonomy Mining (PTM) algorithm to better derive and understand the concepts in the user's query. They used a training document that was designed to derive different patterns. Babashzadeh, Daoud, & Huang, 2013 has proposed an association rule mining (ARM) technique for an IR task in which the set of concepts is derived from the user query and the relationship established by ARM between these concepts. Pattern term mining algorithm reduces noise by deriving closed frequent items and compares these patterns to user query.

El Mahdaouy, El Alaoui, and Gaussier (2018) explored neural word embedding models that represent words in a low-dimensional vector space model to deal with the discrepancy of features. They described the word embedding similarity technique with the probabilistic IR model for the extraction of Arabic information. A document-based neural relevance model (DNRM) is presented in Ran, He, Hui, Xu, & Sun, 2017 to address the mismatch problem of query-relevant documents. A feed-forward network is set up to capture the relationship between the documents relevant to the query and the set of pseudo-relevant documents. Interpolation with the BM25 score technique is used to boost the performance of the DNRM model. The main limitation of this work is the initial set of the required document-query pair as input to the DNRM model.

Song, Hu, He, & Dou, 2019 discussed the effect of density function on proximity-based information retrieval. Because in healthcare domain terms have different influence distribution in different circumstances, a self-adaptive approach on the Saturated Density Function (SDF) is built with the Gamma process. It estimates the distribution of each term and helps to achieve a density-based probabilistic weighting retrieval model. However, the work has just considered document level influence of term's density distribution and not the sentence or a passage level.

2.2. Cluster based approaches

A proximity-based clustering approach-CAWP to correctly capture the clustering structure of a large corpus is studied by Mei and Chen (2014). The authors have identified representative documents from each cluster instead of searching the entire cluster space for each query. Naini, Altingovde, and Siberski (2016) presented a cluster-based greedy local search (C-GLS) algorithm and studied diversification of search results on distributed setup using K-means clustering for large datasets to keep indexing on multiple nodes. The authors have first used the K-means algorithm to get clusters and then applied the C-GLS algorithm to find out diversified results for a user query. Although this approach has provided better diversification of document space, it has not been able to maintain the quality of results for homogeneous queries. Jin et al., 2016 have prepared an efficient indexing mechanism on clustered documents for Information retrieval. Their approach can only provide a solution to conjunctive queries.

A cluster-based query expansion technique is shown in Khennak, Drias, Kechid, & Moulai, 2019, where the k-medoids clustering algorithm is used to group similar terms from the global vocabulary list, and the K-means algorithm is used to calculate new centroids for each cluster. The most similar cluster is selected for each query term to generate the most appropriate candidate terms based on the distance of the respective cluster centroid. The original query is then expanded to include top-ranked candidate terms for document retrieval.

Paper Sankhavara (2020) discussed an NLP based feature weighting scheme with classification and clustering method for the identifying the relevant feedback documents in query expansion. The authors introduced a modified TF-IDF feature scoring technique based on the semantic type of terms used for document representation in the classification process. However, human intervention is required to extract the initial set of feedback documents in order to learn the classification model.

2.3. Bio-inspired approaches

Several bio-inspired approaches, such as evolutionary approaches (Fan, Gordon, & Pathak, 2004; Jung, 2012; Lin, Chen, & Wu, 2014) and swarm intelligence approaches (Buscher, Dengel, Biedert, & Elst, 2012; Collett, Graham, Harris, & Hempel-de Ibarra, 2006; Djenouri, Belhadi, & Belkebir, 2018; Khennak & Drias, 2016; 2017; Picard, Revel, & Cord, 2012; Sharma, Pamula, & Chauhan, 2019a; Zhang, Mei, Liu, Tao, & Zhou, 2011) have been studied in recent years to explore the solution for information retrieval in different domains.

In paper Chawla (2016), the authors presented a genetic algorithm and a clustering-based personalized web search system, where they first applied clustering to the user's hit URLs and then used a genetic algorithm to generate the score and rank query of the relevant URLs. Khennak and Drias (2016) explored the flashing behavior of fireflies and the phenomenon of bioluminescent communication to improve retrieval efficiency by expanding user queries in the medical domain. Authors have used a firefly algorithm to select the most appropriate query from the candidate set of expanded queries. Djenouri et al. (2018) proposed bees swarm intelligence-based data mining approach for information retrieval. The authors have first applied the K-means algorithm to cluster documents and extract closed frequent patterns (CFP) from these clusters. In a later step, these CFPâs are used to generate matching scores with query CFP and explore clusters based on scores using bees swarm optimization technique.

Extensions of bio-inspired approaches is continuously being studying by researchers. One of such is the Accelerated PSO (APSO), an enhanced Particle Swarm Optimization (PSO) version with more simplicity, global optimization capability, and high flexibility used by multiple researchers for query reformulation. Authors in Khennak and Drias (2018) presented the study on a fuzzy K-means to make clusters with similar characteristics terms and combined the Firefly algorithm with an accelerated PSO to extract expansion candidate terms from the clusters for query expansion. Paper Sharma et al. (2019a) proposed an optimized query expansion technique based on a nature-inspired cuckoo search and a fuzzy logic enhanced APSO algorithm to extract relevant expanded query. More precisely, the authors focused on selecting an expanded query from the pool of possible queries rather than any specific expansion term. Approaches based on APSO are also studied in papers (Gupta & Saini, 2017; Khennak & Drias, 2017; Sharma, Pamula, & Chauhan, 2019) to improve the quality through the fuzzification process.

Several researchers have studied the applications of the collective behavior of bird-flocks. Flock based gravitational search algorithm (BFGSA) is studied to improve the results of the basic gravi-

Table 1
Summarization of Pattern Mining IR Approaches.

Author	Description	Limitation
(Beil, Ester, & Xu, 2002)	Hierarchical Frequent Term based Clustering (HFTC) is proposed to get frequent patterns and clusters are formed.	Generates a large amount of candidate set thus, consumes more memory for large datasets.
(Fung, Wang, & Ester, 2003)	Frequent itemset based hierarchical clustering approach is proposed.	Requires longer time for execution and consumes more memory.
(Yu, Searsmith, Li, & Han, 2004)	Presented a topic modelling based clustering for IR.	Fails to deal with overlapping clusters for highly correlated documents.
(Kontostathis, 2007)	Proposed an optimized LSI model to retain dimensions which contain term relationship and combined it VSM for document IR	Document representation gets complicated with increasing dimensions for large datasets.
(Zhong et al., 2012)	Proposed a Pattern Taxonomy Mining (PTM) algorithm to derive and understand concepts in users query more correctly.	Prior knowledge about the query relevant document is required to learn concepts.
(Babashzadeh, Daoud, & Huang, 2013)	Association rule mining technique is proposed to determine the relationship between query concepts and document set.	The proposed approach is memory inefficient.
(Marchesin, Purpura, & Silvello, 2019; Ran, He, Hui, Xu, & Sun, 2017)	Proposed a neural word representation based PRF model to establish a semantic association between documents and query terms.	A relevant query-document pair is required initially. Dataset is required to be scanned twice, hence not feasible with large datasets.
(Song, Hu, He, & Dou, 2019)	A proximity-based self-adaptive Saturated Density Function for IR is proposed.	The influence of the term's density distribution is not considered on sentence or passage level.

Table 2
Summarization of Cluster Based IR & Bio-Inspired IR Approaches.

Author	Description	Limitation
Cluster Based IR Approaches		
(Mei & Chen, 2014)	Proximity-based clustering approach is presented to identify representative documents from each cluster to match with user query.	No. of objects and corresponding weights get increased with cluster count.
(Naini et al., 2016)	K-means cluster-based greedy local search (C-GLS) algorithm is presented to find out diversified results for the user queries.	Poor result quality for homogeneous queries.
(Jin et al., 2016)	Presented an efficient indexing mechanism on clustered documents for IR.	The solution is provided only to conjunctive queries.
(Khennak, Drias, Kechid, & Moulai, 2019)	A combined K-medoids and K-means clustering is used to find the most similar query terms for query expansion.	Local minima problem occurred due to K-means is not addressed.
(Sankhavara, 2020)	NLP based weighting scheme with classification and clustering is presented to determine relevant feedback documents.	The initial set of feedback documents are generated with human knowledge.
Bio-Inspired IR Approaches		
(Chawla, 2016)	Presented genetic algorithm(GA) and clustering based personalized web search system on URLs. GA is used to generate score and rank query relevant URLs.	Fails to classify all the web pages based on some key features or any specific link towards each web page
(Khennak & Drias, 2016)	Explored the flashing behavior of fireflies in query expansion to select the best-suited query from the candidate set of expanded queries.	Two phase retrieval process with longer retrieval time.
(Han et al., 2017)	Studied flock based gravitational search algorithm (BFGSA) for diversely searching results.	For large datasets, this approach generates huge candidate population with initial search results.
(Djenouri et al., 2018)	Proposed a bees swarm intelligence-based data mining approach for IR. Frequent patterns are extracted from clusters formed with K-means.	Local minima problem due to K-means is not addressed.
(Khennak & Drias, 2018)	A combined approach based on APSO and fuzzy K-means is presented for query expansion.	Local minima problem due to K-means is not addressed.
(Sharma et al., 2019a)	APSO based optimization with cuckoo search technique is proposed for query expansion.	With the increasing dataset size it becomes more complex.

tational search algorithm (GSA) in Han, Qiang, and Lan (2017). The candidate population is initially generated for diverse search results, and positions are determined by neighbors. It is found that the application of flock-based technology to GSA has enhanced search performance. Paper Biswas and Pal (2019) discussed an interesting application of the Genetic Algorithm (GA) along with a priority-based fuzzy goal programming method to address congestion management problems in power transmission lines. The PSO-based heuristic approach is presented in Ganguly (2020). It helps to assess the optimum value of the penetration level of the Distributed Generation and the network performance index in order to achieve the optimum sites and sizes of DG divisions. In particular, heuristic algorithms are expected to optimize the number of decision variables and different constraints. Tables 1 and 2 provides the brief summary of IR approaches studied in recent years.

Several retrieval frameworks have been studied in recent years; most of the bio-inspired approaches have focused on query reformulation, which may lead to delayed results. The proposed IR framework is an attempt to improve the IR results by integrating the swarm intelligence with memory efficient data mining techniques. We performed document clustering where documents are considered to be an object in the virtual search space and the movements of these objects are governed by four heuristic rules. In addition, the fuzzy nature of the clusters helps to measure the relevance of each document without missing a single instance. We employed memory efficient pattern mining technique to extract

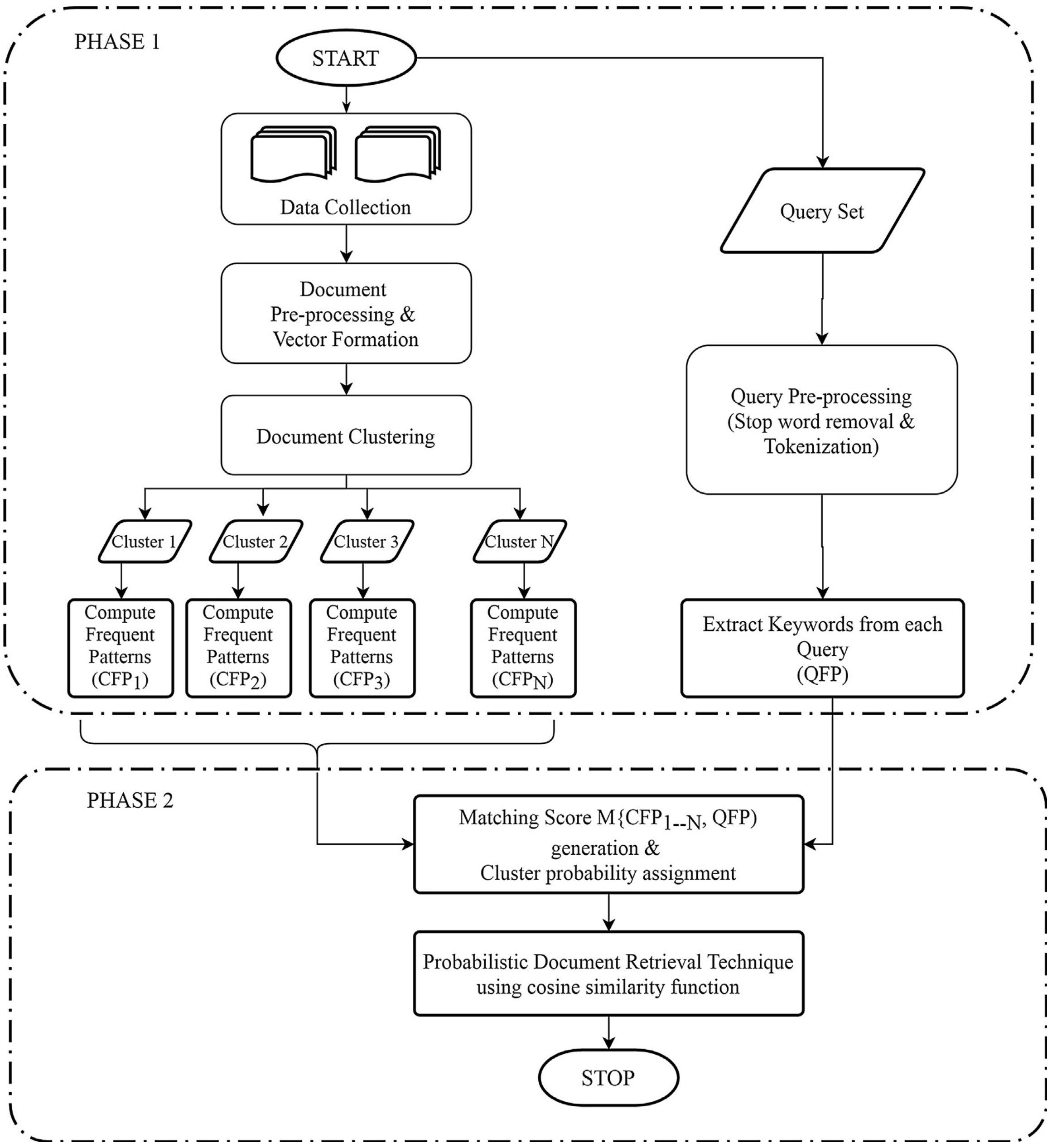
PHASE 1
START
Data Collection
Document Pre-processing & Vector Formation
Document Clustering
Cluster 1
Cluster 2
Cluster 3
Cluster N
Compute Frequent Patterns (CFP1)
Compute Frequent Patterns (CFP2)
Compute Frequent Patterns (CFP3)
Compute Frequent Patterns (CFPN)
Query Set
Query Pre-processing (Stop word removal & Tokenization)
Extract Keywords from each Query (QFP)
PHASE 2
Matching Score M{CFP1--N, QFP} generation & Cluster probability assignment
Probabilistic Document Retrieval Technique using cosine similarity function
STOP

Fig. 1. The proposed IR architecture.

closed frequent patterns from the fuzzy clusters and this helps to generate matching scores for clusters with specific queries. These matching scores are then used in probabilistic document retrieval technique to obtain relevant documents for the query.

3. The methodology

Fig. 1 shows the flow of the proposed framework, and each component is discussed in detail as below:

3.1. Document preprocessing and vector modeling

Text documents extracted from corpus contain low-value words, noisy data, and therefore, to eliminate it, we applied natural language processing techniques (NLP) (Loper & Bird, 2002) on each document. We used a modified English language stop-word list to exclude low-value terms such as *a, am are, be, before, us, use, who, thus,* etc., as these since these words contribute very less in the overall weighting technique. We then tokenized each document to form the bag-of-word model, which is considered as the global vocabulary of terms from the corpus. Since TREC-CDS and OHSUMED data collections contain medical terms hence to preserve their importance, stemming is not performed on words to get its root form. Once the global vocabulary is created, a document vector i.e., a vector space model (VSM), is designed, which contains the count of occurrences of each term in the specific document.

These document vectors are further used for the term weighting technique. Term weighting techniques assign scores to terms in the document vector that helps to ascertain term importance in the corpus. Out of several weighing schemes available in state-of-art (Salton & Buckley, 1988) term frequency inverse document frequency (Tfidf) has outperformed others. It can be calculated as in Eq. (1).

$$Tf - Idf(i, |d) = (\sqrt{wf_{id}})Ln\left(\frac{N}{df_i}\right) \quad if \quad wf_{id} \geq 1 \tag{1}$$

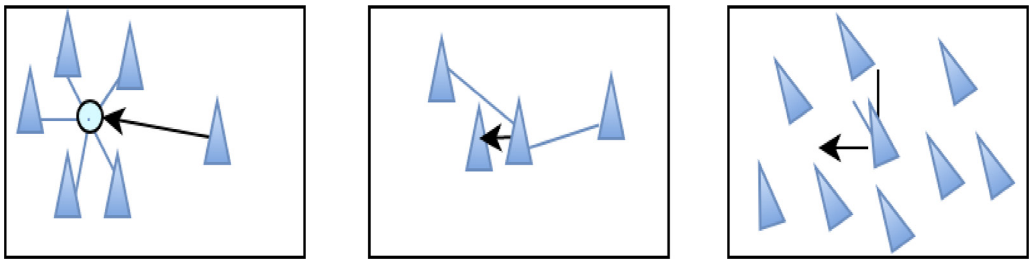

Fig. 2. Basic rules of boid movement (a)Cohesion (b)Separation (c)Alignment.

Where *Tf* stands for term frequency and *idf* is the inverse document frequency calculated as the number of documents containing a particular word.

For large datasets such as TREC-CDS and OHSUMED, which produce a massive number of features, it is difficult to process high dimensional document vectors; therefore, we used dimensionality reduction latent semantic analysis (LSA) technique (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990; Landauer, Foltz, & Laham, 1998). Semmarization of Pattern Mining IR Approadiscrete topics or concepts explained in the document. It is reasonable that the number of concepts is always much lower than the total number of features. In LSA, the original document-term matrix is decomposed into a small matrix using the singular value decomposition technique (SVD). Although LSA discards some characteristic features and may suffer retrieval precision, but it is a technique that replaces a set of co-occurring terms into concepts that are used in the same context for several terms. That is why it helps to calculate similarity in latent semantic space.

3.2. The proposed bio-inspired K-Flock clustering algorithm

Bio-inspired swarm intelligent (Bonabeau, Dorigo, & Theraulaz, 1999) models such as particle swarms, ant colonies, bird flocking behavior, fish school, bee warms, etc. are all agent-based models used in the number of optimization tasks. As compared to individual agents, the collectively exhibited complex and emerging behavior of multiple search agents are more robust and resistant to failure. Such optimization algorithms are globally characterized by changing the environment of a large number of agents. We adopted stochastic and heuristic principle discovered from observing bird flocks or fish schools for clustering text documents. The Flocking model was first devised by Reynolds (1987) to simulate the collective behavior of boids, i.e., bird-like objects on computer graphics-based animation, as a study of emergent behavior achieved collectively without a global leader. In a flock based clustering model, the movement of each object, i.e., boid, is governed by three steering rules suggested by Reynolds viz. Cohesion, Alignment, and Separation. In addition to these, we applied the fourth rule for document clustering, i.e., document similarity dissimilarity rule. We assume each document vector as a boid *B* and represented in a 2D virtual space with position L_b, surrounded by *X* other L_x positioned vectors. This representation is very much similar to real flocks; the boids with similar characteristics will form a group together, while those with different features will repulse each other. The movement of these boids is governed by above mentioned four steering rules and are described below:

3.2.1. Cohesion

This rule states that each boid will force itself to move towards the average local center, i.e., centroid. This movement helps to keep *n* similar adjacent objects together. Fig. 2(a) shows the movement of boid by cohesion rule w.r.t other particles. In the 1 algorithm, line 14–16 describes that the boid position has changed w.r.t to the local average value, i.e., local centroid position L_c. Cohesion can be calculated as in Eq. (2).

$$L_b = L_b + \left[\frac{1}{n}\sum_{x=1}^{n} L_x\right] - L_c \tag{2}$$

3.2.2. Separation

It gives the boids the potential to keep neighboring particles away from each other so that no collision occurs within the same group. This repulsive force of boids helps to avoid overcrowding. As shown in Fig. 2(b), the boids within a specific range are considered. In order to apply the separation rule for document objects, the fitness value of each document object with its adjacent agents is calculated using the Schaffer F6 function, as shown in the Eq. (3). Here *x* and *y* are dimensions of each document vector. This function helps us to get the highest scoring boid as a new local centroid, and it separates low scoring boids from the neighboring boids.

$$f(x,y) = 0.5 + \frac{sin^2(x^2 - y^2) - 0.5}{(1 + 0.001(x^2 + y^2))^2} \tag{3}$$

3.2.3. Alignment

As shown in Fig. 2(c), this rule attempts to align the boid vector with its average local neighborhood vector, which leads to all objects heading in the same direction. In reality, it is the matching of the velocity of one object with the average velocity of other flockmates. As in document clustering, the document acts as a boid; therefore, this rule is combined with the cohesion rule. For N local boids within radius R with centroid c and velocity v_c and v_x as the velocity of adjacent boids, the change in alignment value v_b is mathematically calculated as in Eq. (4).

$$v_b = \left[\frac{1}{n}\sum_{x=1}^{n} v_x\right] - v_c \tag{4}$$

3.2.4. Similarity & Dissimilarity rule

This rule is designed solely to enhance the capability of the basic flock model for document clustering; it considers Tfidf vectors to group similar document objects together and separates the dissimilar one. As shown in line number 19–21 of the Algorithm 1, the distance between two document objects is calculated using the well-known cosine similarity measure, and objects are moved accordingly. Mathematically, the distance between two document objects *D*1 and *D*2 is calculated using the Eq. (5) shown below-

$$cos(\theta_{D1,D2}) = \frac{\sum_{i=1}^{n} D1_i \ D2_i}{\sqrt{\sum_{i=1}^{n} D1_i^{\,2}}\sqrt{\sum_{i=1}^{n} D2_i^{\,2}}} \tag{5}$$

The steps in the K-Flock algorithm are presented in Algorithm 1 and its description can be summarized as follows:

Algorithm 1 Bio-inspired K-Flock Clustering Algorithm.

```
Input: VSM: Document vector; K: cluster count; MAX-ITERATION = 10.
Output: C= {$C_1, C_2, \ldots, C_k$} document vectors decomposed into K clusters.
Variables: M: total count of document vectors; C: cluster vector;
    centroid, centroidNew: cluster centroid list;
    VSMFit: A list for document fitness values;
    VSMBestFit: A list for best fit document vectors of each cluster.
Begin
 1: Initialize K random centroids from document vectors as $centroid[1.K]$
 2: for each $centroid[i]$ do
 3:     Generate initial clusters for centroids in $centroid[1.K]$ using eq. 5
 4: end for
 5: Now apply heuristic rules to generate final clusters.
 6: while $MAX - ITERATION$ do
 7:     for each cluster $C[i]$ do
 8:         for each document vector $VSM[j]$ in cluster $C[i]$ do
 9:             Evaluate vector fitness value w.r.t $centroid[i]$ using Schaffer's F6 function and store values in $VSMFit$.
10:         end for
11:         for each cluster $C[i]$ do
12:             Compute $bestfit$ using $VSMFit$ and store in $VSMBestFit[i]$.
13:         end for
14:         for each document vector $VSM[j]$ in cluster $C[i]$ do
15:             Accelerate the document vector w.r.t its $bestfit$ vector.
16:         end for
17:         Now new centroids $centroidsNew[i]$ are the $VSMBestFit[i]$.
18:     end for
19:     for each $centroidNew[i]$ do // compute new clusters
20:         Compute new clusters $CNew[i]$ using score in equation.5.
21:     end for
22:     if $centroid == centroidNew$ then break;
23:     else $centroid == centroidNew$
24:     end if
25: end while
End
```

The input to the algorithm is document vectors containing Tfidf weights. We also provided the maximum number of iterations allowed and the cluster count, i.e., K. In the output, the algorithm is expected to produce K fuzzy clusters designed using the rules as mentioned above. The fuzzy characteristic of the cluster helps to measure the relevance of each document without missing a single instance.

The algorithm starts with the initialization of random values for the centroids (line 1). Clusters are formed by measuring the distance between other document vectors and the initial centroids using cosine similarity defined in Eq. (5) (lines 2–4). Four heuristic rules are applied to the movement of initial cluster members to achieve the best solution and are updated in each iteration. Lines 6–25 of the 1 algorithm describes how these heuristics rules are used to govern the movement of document objects. Lines 7–10 defines the movement of objects governed by the separation rule where the Schaffer's F6 function generates the fitness value for each document object w.r.t cluster centers. The best document objects from each cluster are then computed using these fitness values (lines 11–13). The Cohesion and Alignment rules are jointly imposed on objects as discussed earlier and are defined on lines 14–16. Newly derived best document objects are considered to be new centroids and used for assignment of document objects to form new groups. This process is governed by a document similarity &dissimilarity rule, i.e., using a cosine similarity measure. The algorithm stops when it reaches the maximum number of iterations or when it converges to the optimal solution.

3.3. Frequent Pattern Mining

Frequent Pattern Mining (FPM) technique helps to gain more insight through the context of a dataset in less time. It is a widely used technique to discover interesting patterns, concepts from the input data.

In recent years, the approach to mining patterns has changed; various techniques based on high utility and constraint are designed to study patterns. High utility based mining algorithms consider an item in frequent or infrequent patterns, no matter if it is present (positive patterns) or absent (negative patterns) in the data. Real life approaches such as a market basket analysis to generate high profit (Fournier-Viger, Lin, Nkambou, Vo, & Tseng, 2019) or low profit even though patterns are frequently purchased (Gan et al., 2018) are presented. In other words, the concept of high-utility patterns indicates that not all items are always equally important. Although these approaches have shown improvement over previous work, they still generate a large number of candidates. Since these techniques rely on the loose upper bound of the candidate's utility and scan the original database multiple times to update high utility items, it takes longer running time and consumes high memory.

In Abboud, Brun, and Boyer (2019); Van, Vo, and Le (2018), several constraint-based techniques such as robustness, closedness, and extended closedness, etc. are investigated. Constraint-based approaches are prominent on sequence databases to generate patterns as needed by the user. Sequence databases, however, are often large and contain noisy data. As a result, the algorithm exhibits poor performance due to enormously large search space; it generates large number of candidates, particularly with low support threshold. Although several constraints (Abboud et al., 2019) help to reduce large search space at the very beginning of the mining process, there is a chance that valuable information will be lost if the relevant patterns are discarded. Also, with the varying constraint, the entire sequence database needs to be retracted. If this happens frequently, the algorithm consumes more memory and takes a longer time to execute. Moreover, all the above algorithms suffer from the same limitations as Apriori-based ARM algorithms.

In this work, every single document is considered as a transaction, therefore more often, these transactions are sparse, which may lead to higher computational costs and discover poor patterns. Therefore the approaches discussed above enumerate more patterns, especially when the minimum support is low. In order to deal with this situation, we used relative minimum support and employed memory efficient frequent itemset mining Recursive Elimination (RElim) Algorithm (Borgelt, 2005). RElim works more prominently on sparse data; i.e., it uses depth-first-traversal to generate a candidate set at once and maintain only one transaction per item. Some commonly used terms in the FPM are explained as follows:

- *Itemset*: It is the collection of one or more words derived from document.
- *Support*: It is the ratio of number of transactions with itemset to total transactions. It is defined using the Eq. (6).

$$Supp(A \Rightarrow B) = \frac{|\{t \in T | A \cup B \subset t\}|}{|T|} \tag{6}$$

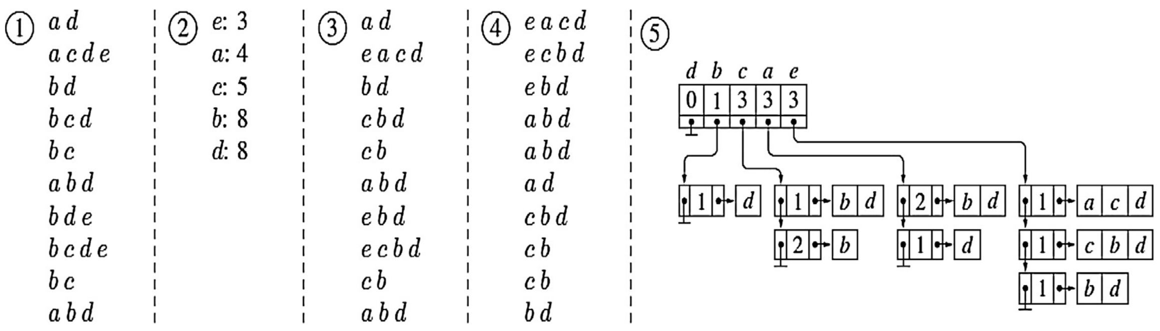

Fig. 3. RElim Transaction database pre-processing.

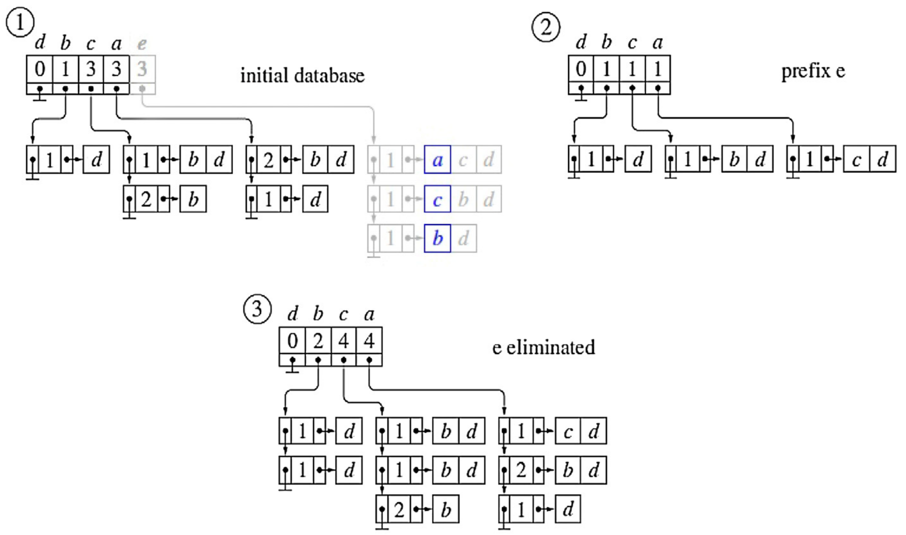

Fig. 4. RElim algorithm steps.

For large datasets such as TREC-CDS and OHSUMED, transactions are more sparse, so we used relative minimum support $s_{\min}$ count for each cluster C_i and are calculated as in Eq. (7), and for other datasets, we considered $s_{\min} = 50\%$.

$$s_{min}[i] = s_{min} * \frac{|C_i|}{m} * a \tag{7}$$

where a is some constant and m is total number documents in the collection.

- *Frequent Pattern (P)* The transaction database contains itemsets. Hence the occurrences of P in transaction denoted by $T(P)$ has frequency $\geq MSupp$ i.e., minimum support is considered as the frequent pattern.
- *Closed Frequent Pattern* A pattern P is closed frequent pattern if there does not exist any subset of P.

RElim algorithm is implemented for each document cluster *Ci* to generate closed frequent patterns. The pseudo-code is described in the Algorithm 2. The initial database preparation of the RElim algorithm is explained in Fig. 3. In Step 1, a transaction database is extracted. In step 2, the frequency of each item is counted, and items having a frequency less than minimum support are eliminated. In step 3, frequent itemsets are sorted in descending order followed by lexicographic sorting of transactions in step 4. The database prepared is shown in step 5.

Fig. 4 gives the pictorial view of Algorithm 2: in step 1, the least frequent itemset *e* is processed. If the count for *e* in list is greater than $s_{\min}$ then *e* is treated as frequent itemset. Step 2 shows the trailing itemsets of *e*, which are recursively processed same as in step 1. In step 3, the itemset list obtained in step 2 will be added back to the original conditional database at their respective succeeding items. This original conditional database is recursively processed to extract all other frequent itemsets. We applied *SET* operation to remove subset patterns and considered only closed frequent patterns.

3.4. Query key-phrase extraction

A key-phrase extraction is a three step process, as shown in Fig. 5. Preparing a candidate keyword set is the first step in which all words, phrases, and concepts that could possibly be a keyword or part of a keyword are extracted, and other low-value terms such as punctuations, spaces, and numbers are removed. As shown in the Algorithm 3, we considered every adjoining word separated by whitespace as a keyword and split the sequence at every stop-word occurrence. The second step is to identify discriminating features by calculating its frequency, probability, or degree of the word, i.e., the total count of candidate keywords that contain a word. We calculated the score for each word, as shown in Eq. (8) below:

$$S(W_i) = \frac{\sum_{j=1}^{kc} count\ of\ words\ in\ (K_j)\ containing\ W_i}{F(W_i)} \tag{8}$$

Where kc is the total number of keywords, $F(W_i)$ is the frequency of word.

The third step is to rank and select keywords. Keywords are ranked by summing up the score of all words they contains. We extracted one-third of the top score keywords as a final key-phrase for the query document.

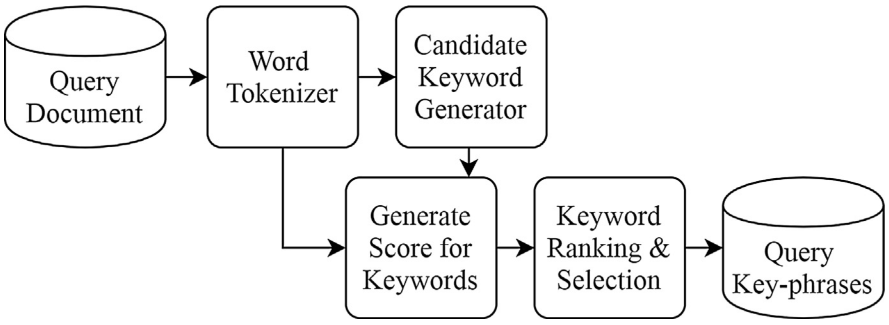
Query Document
Word Tokenizer
Candidate Keyword Generator
Generate Score for Keywords
Keyword Ranking & Selection
Query Key-phrases

Fig. 5. Key-phrase Extraction Process.

Algorithm 2 RElim Algorithm.

```
Input: $D$: conditional database; $p$: prefix item in D; $s_{\min}$ :minimum support.
Output: Frequent Patterns
Variable: $txnList$: array of transaction lists for current item
          $itemList1$, $itemList2$: transaction list elements to traverse
Begin
 1: itemcount = 0;
 2: while D has items do
 3:     l = length(D) - 1; s = D[l].wt //get support of item from D.
 4:     if $s \geq s_{\min}$ then
 5:         $pre = pre \cup item(l)$;
 6:         report the frequent item set found with support s.
 7:         create empty transaction list txnList of $l-1$ items; itemList1 = D[l].head;
 8:         while itemList1 do
 9:             itemList2 = copy of itemList1 ; itemList1 = get next successive item; k= itemList2.items[0];
10:             remove k the leading item from the copy itemList2.items
11:             if itemList2 is not empty then
12:                 itemList2.succ = txnList[k].head; txnList[k].head = itemList2;
13:                 txnList[k].wt += itemList2.wt;
14:             end if
15:         end while
16:         itemcount += RElim(txnList,pre,$s_{\min}$) + 1 // recursively process.
17:         pre -= item(l);
18:     end if
19:     itemList1 = D[l].head;
20:     while itemList1 do
21:         itemList2 = itemList1; itemList1 = get next successive item;
22:         k = itemList2.items[0];
23:         remove k the leading item from the copy itemList2.items
24:         if itemList2 is not empty then
25:             itemList2.succ = D[k].head; D[k].head = itemList2;
26:             D[k].wt += itemList2.wt;
27:         end if
28:     end while
29:     remove D[l] from D;
30: end while
End
```

Algorithm 3 Pseudo-code for Query Key-phrase Extraction.

```
Input:    $Q$: Query Document.
Output:   K = $\{K_1, K_2, \ldots, K_n\}$ Keyword List.
Variable: kc: keyword count; F[$W_i$]: frequency of word;
          R[$K_i$]: Rank of keyword
Begin
 1: Q = $[W_1, W_2, \ldots, W_n]$ Split query document into a sequence of tokens i.e. array of words considering each word as a candidate keyword.
 2: for each word $W_i$ do
 3:     if $W_i$ NOT stop-word then
 4:         $K_j = K_j + W_i$
 5:     else
 6:         j++ // split the sequence to get new keyword
 7:     end if
 8: end for
 9: kc = j
10: for each word $W_i$ do
11:     $S(W_i) = \frac{\sum_{j=1}^{kc} count\ of\ words\ in\ (K_j)\ containing\ W_i}{F(W_i)}$
12: end for
13: for each keyword $K_j$ do
14:     $R(K_j) = \sum_{i=1}^{n} S(W_i) \in K_j$
15: end for
16: Select one-third top scoring candidate keywords as the extracted keywords.
End
```

3.5. Probabilistic Document Retrieval Model (PDRM)

The document retrieval task is divided into two steps, i.e., the first step is to assign probabilities to clusters, and the second step is to retrieve assigned probability value documents from each cluster for queries.

3.5.1. Matching function and probability assignment

As shown in Algorithm 4, the probability of selecting each cluster is calculated by matching it's closed frequent patterns with key-phrases extracted for the query. At first, we computed the number of common terms per cluster and then calculated the probability of consideration using the total number of terms present in that query. The probability values assigned to each cluster gives the total number of documents that are required to be extracted from each cluster.

Let Q be the query with set of k terms as $Q = \{t_1, t_2, \ldots, t_k\}$ and key-phrases for query as $QFP = \{\{t_1, t_3\}, \{t_3, t_4, t_6\}, \{t_8, t_{10}\}\}$ with total 6 terms

Also, let's have 3 clusters $C = \{C_1, c_2, c_3\}$ and 10 documents which are decomposed in each clusters as follow:

- $C_1 = \{d_1, d_3, d_4\}$
  with the frequent patterns $CFP_1 = \{\{t_2, t_3\}, \{t_1, t_3, t_4\}, \{t_{10}, t_{11}\}\}$
- $C_2 = \{d_2, d_6, d_7, d_10\}$
  with the frequent patterns $CFP_2 = \{\{t_1, t_2\}, \{t_5, t_7, t_{11}\}, \{t_{12}\}\}$
- $C_3 = \{d_5, d_8, d_9\}$
  with the frequent patterns $CFP_3 = \{\{t_5, t_6\}, \{t_8\}\}$

The matching score M between CFP_i and QPF is calculated as follow:

$$M(CFP_i, QFP) = \left| \bigcup_{i}^{|CFP_i|} (CFP_j^i \cap QFP) \right| \tag{9}$$

Where n is the number of closed frequent patterns in cluster C_i and CFP_j^i is the j^{th} frequent pattern in of cluster C_i.

Considering the above case of 10 documents and 3 clusters we will have 3 matching scores M_1, M_2, M_3 for 3 different clusters with-

Algorithm 4 Pseudo-code for PDRM.

```
Input: D : Document vector; Q : Query vector;
set of k clusters of documents, $C = \{C_1, C_2, \ldots, C_k\}$;
set of closed Frequent patterns of k clusters $CFP = \{CFP_1, CFP_2, \ldots, CFP_k\}$;
Key-phases for query Q, $QFP = \{QFP_1, QFP_2, \ldots, QFP_m\}$;
$N$ : Number of documents to be retrieved for query Q.
Output: Relevant document list for query $Q$
Variable: kc: keyword count; F[$W_i$]: frequency of word; R[$K_i$]: Rank of keyword
Begin
1: for each cluster $C_i$ do
2:     Calculate matching scores of closed frequent patterns.
       $M(CFP_i, QFP) = \left| \bigcup_i^{|CFP_i|} (CFP_j^i \cap QFP) \right|$
3: end for
4: For every cluster $C_i$ calculate probability values
   $P_i = \frac{M_i}{\sum_i^n M_i}$
5: for clusters $C_i$ having $P_i > 0$ do
6:     Calculate similarity value between document and query vector.
       $cos(\theta_{D_i,Q}) = \frac{\sum_{i=1}^n D_i \; Q}{\sqrt{\sum_{i=1}^n D_i{}^2}\sqrt{\sum_{i=1}^n Q^2}}$
7:     Retrieve $P_i$% of N documents using cosine similarity function.
8:     Rank selected documents in the descending order of similarity score.
9: end for
End
```

$QFP = \{\{t_1, t_3\}, \{t_3, t_4, t_6\}, \{t_8, t_{10}\}\}$ as

- $M_1 = |(CFP_1^1 \cap QFP) \cup (CFP_2^1 \cap QFP) \cup (CFP_3^1 \cap QFP)| = |\{t_1, t_3, t_4, t_{10}\}| = 4$
- $M_2 = |(CFP_1^2 \cap QFP) \cup (CFP_2^2 \cap QFP) \cup (CFP_3^2 \cap QFP)| = |\{t_1\}| = 1$
- $M_3 = |(CFP_1^3 \cap QFP) \cup (CFP_2^3 \cap QFP)| = |\{t_5, t_6\}| = 2$

Likewise the probability P_i of selecting number of documents from each cluster can be given as in Eq. (10)

$$P_i = \frac{M_i}{\sum_i^n M_i} \tag{10}$$

- P_1= 4/7 = 57% of total number of documents to be retrieved will come from cluster C_1
- P_2 = 1/7 = 14% documents will be retrieved from cluster C_2
- P_3 = 2/7 = 29% documents will be retrieved from cluster C_3

3.5.2. Document retrieval

The last step in the proposed framework is to retrieve documents on the basis of the assigned probabilities. Traditionally, a similarity between the query vector and the document vector is generated in the retrieval process. Based on the score, the relevance of the document is decided. We represented vectors with Tfidf weights and used a well-known cosine similarity function to evaluate the similarity between these vectors. The similarity value of $S(D_i, Q_i)$ can be calculated as shown in Eq. (5). For two vectors, if they have 0^o degree angle between them, i.e., if two vectors are parallel to each other and growing in the same direction, then cosine function will generate a value of 1. For vectors that are perpendicular to each other, i.e., the angle between two vectors is 90^o; then, the function generates a value 0. We retrieved P_i% of total documents from the clusters and ranked them in the descending order of their similarity scores.

4. Experimental details

4.1. Dataset collection

We have extensively conducted experiments on five different size datasets:

1. *TREC-Clinical Decision Support Trac 2014-15*[1] TREC is a very large corpus consist of 733,138 full text articles in NXML format, which are drawn from the open access subset of PubMed Central repository. It is used in TREC-CDS 2014 and 2015 tracks with a different set of queries.
2. *OHSUMED*[2] is a large size collection of 348,566 records (split as a training- 54,710 records and testing- 293,856 records) collected from Medline- an online medical information database. It contains title, abstract, and MeSH indexing terms from 270 medical journals over a period of five years(1987-1991). It has 63 queries and relevance judgments.
3. *NPL*[3] is a medium-sized document collection of 11,429 records containing the title and abstracts of the physical laboratory experiments.
4. *LISA*[3] is another medium-sized corpus with 6004 records taken from the library and information science abstracts.
5. *CACM*[3] is a medium-sized document corpus with a collection of 3204 HTML formatted documents published in the Communications of the ACM Journal between 1958 and 1979. Each document contains the title, abstract, and details of the author.

We used the respective parsers to extract the content of each document and converted it into a text document for further use. Each dataset consists of a set of queries and their relevant ground truths prepared by experts. All algorithms are implemented, and the experiments are performed in python. Table 3 provides brief statistics on datasets.

4.2. System effectiveness measures

In this section, we describe the evaluation measures used to measure cluster accuracy and the metrics used in the assessment of the IR system. We also discussed the selection of parameters based on some initial set of experiments.

4.2.1. Cluster performance evaluation

We performed clustering on datasets for which ground truth labels are unknown. As the clusters formed are fuzzy, the basic silhouette coefficient index could not generate a valid index score. Thus, to measure the performance of clusters, we used a modified silhouette coefficient index measure. In the paper (Rawashdeh & Ralescu, 2012), the generalized intra-inter silhouette index is introduced to evaluate the performance of such fuzzy clusters. Eq. (11) gives the basic version of silhouette index for datasets containing n objects as follows:

$$S = \frac{1}{n}\sum_i^n \frac{b_i - a_i}{max(a_i, b_i)} \tag{11}$$

Here, a is the mean distance between the object and the other points belonging to the same cluster; b is the mean distance between an object to other points of the nearest cluster.

There are two components considered in the evaluation: compaction within cluster and separation amongst clusters. Cluster quality is labeled as good if it has minimum intra-distance in within class objects and maximum inter-distance between objects

[1] http://www.trec-cds.org/2014.html.
[2] https://trec.nist.gov/data/t9_filtering.html.
[3] http://ir.dcs.gla.ac.uk/resources/test_collections/.

Table 3
Dataset Statistics.

Dataset Name	Document Type	No. of Documents	No. of Queries
TREC-CDS 2014 & 2015	Pubmed Articles	733,138	30 (each year)
OHSUMED	Medline Articles	348,566	63
NPL	National Physical Laboratory Experiments Title & Abstract	11,429	93
LISA	Library and Information Science Abstracts	6004	35
CACM	ACM Journal Papers	3204	64

Table 4
Silhouette coefficient index generated on NPL, LISA, CACM dataset with varying cluster size(K).

Dataset	Clustering Algorithm	K = 5	K = 10	K = 15	K = 20	K = 25	K = 30
NPL	K-Flock	**0.6389**	0.5784	0.5006	0.4920	0.4825	0.4920
	K-means	**0.5760**	0.4275	0.4165	0.3852	0.3695	0.3712
LISA	K-Flock	**0.5394**	0.5130	0.5213	0.4908	0.4884	0.4861
	K-means	**0.5197**	0.4126	0.3841	0.3798	0.3427	0.3689
CACM	K-Flock	0.6195	**0.6624**	0.5765	0.5216	0.5230	0.5074
	K-means	0.5295	**0.5368**	0.5103	0.4857	0.4615	0.3938

Table 5
Silhouette coefficient index generated on TREC-CDS 2014 dataset and OHSUMED-Train dataset with varying cluster size(K).

Dataset	Clustering Algorithm	K = 10	K = 20	K = 30	K = 40	K = 50
TREC-CDS 2014	K-Flock	**0.5845**	0.5227	0.5584	0.4912	0.4674
	K-means	**0.4973**	0.4595	0.4617	0.4289	0.4123
OHSUMED-Train	K-Flock	**0.6534**	0.6062	0.5716	0.5529	0.5501
	K-means	**0.5967**	0.5738	0.5496	0.5526	0.5245

of other classes. As discussed in Rawashdeh & Ralescu, 2012 intra-distance and inter-distance are calculated as in Eqs. (12) and (13)-

Intra-matrices w.r.t. each cluster i:

$$IntraDist_i = \left\{ \left[min(u_{ij}, u_{ik}) \right] | i = 1, \ldots, c \right\} \tag{12}$$

Inter-matrices w.r.t. each cluster r and s:

$$InterDist_{rs} = \left\{ max\left[min(u_{rj}, u_{sk}), min(u_{sj}, u_{rk}) \right] | r, s = 1, \ldots, c; r < s \right\} \tag{13}$$

Here, U_{mn} defines the membership value of n^{th} object in m^{th} cluster $|m = i, r, s \;\; \& \;\; n = j, k$.

Now, compaction a and separation b for each object j is computed as:

$$a_j = \min\left\{ \frac{\sum_{k=1}^{n} IntraDist_i(j,k).d_{jk}}{\sum_{k=1}^{n} IntraDist_i(j,k)} | i = 1, \ldots, c \right\} \tag{14}$$

$$b_j = \min\left\{ \frac{\sum_{k=1}^{n} InterDist_{rs}(j,k).d_{jk}}{\sum_{k=1}^{n} InterDist_{rs}(j,k)} | r, s = 1, \ldots, c; r < s \right\} \tag{15}$$

Here, d_{jk} is the distance matrix between objects j and k.

The silhouette coefficient is first calculated for each object, and the average over the complete dataset is the index for the whole system as shown in Eq. (11). Its value varies from -1 to +1; the value towards +1 indicates that clusters are dense and well separated. We measured K-means cluster performance using the basic silhouette coefficient index shown in Eq. (11) and used a generalized intra-inter silhouette index for the proposed K-Flock clustering algorithm. Following Tables 4 and 5 show the silhouette index generated by different cluster sizes for different datasets.

4.2.2. IR System measures

For experimental evaluation, we used trec_eval script, which used the necessary measures to evaluate the IR system against the gold standard query relevant document files, i.e., *qrels* prepared by respective experts for each dataset. In order to measure relatedness, we used the following metrics-

- *Precision (P):* It defines the capability of the system to generate relevant items. Mathematically it can be defined as the fraction of relevant documents extracted from the total number of retrieved documents and can be shown as follows:

$$P = \frac{Relevant\ Documents\ Retrieved}{Total\ Retrieved\ Documents} \tag{16}$$

We computed precision values at a different number of documents retrieved, i.e., P@5, P@10, P@15, and so on. This may help to understand the system's ability to correctly extract items at different volumes.

- *Recall (R):* It gives the fraction of relevant retrieved items to the total number of relevant items present in the corpus.

$$R = \frac{Relevant\ Documents\ Retrieved}{Total\ Relevant\ Documents\ in\ the\ corpus} \tag{17}$$

- *Mean Average Precision (MAP):* Conventional measures such as Precision and Recall are further extended to combinedly build together a more stable and robust measure to evaluate the performance of a system. *MAP* provides the single numeric value across different Recall levels (Croft, Metzler, & Strohman, 2010) for the entire system and all queries. It is the mean of the average precision values calculated for all queries.

$$MAP = \frac{1}{|Q|} \sum_{i}^{|Q|} \frac{1}{n_i} \sum_{j}^{n_i} P(R_{jk}) \tag{18}$$

Here $P(R_{jk})$ is the proportion measured on relevant documents retrieved till the document d_j is observed, |Q| is total number of queries, n_i total relevant documents for q_i.

- *Rprec:* It is the precision calculated after retrieving R relevant documents for any specific query. It is calculated as follows:

$$R - prec = \frac{|r|}{R} \tag{19}$$

Where $|r|$ is the relevant documents retrieved, and R is the total relevant documents per query. Generally, it is said that R-prec is the point where both precision and recall have the same value.

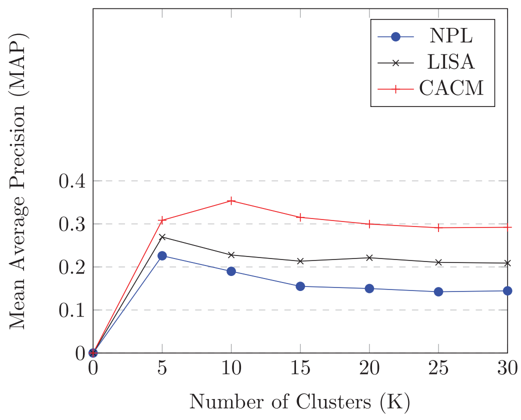

Fig. 6. Quality of documents returned at different cluster number for NPL, LISA, and CACM datasets.

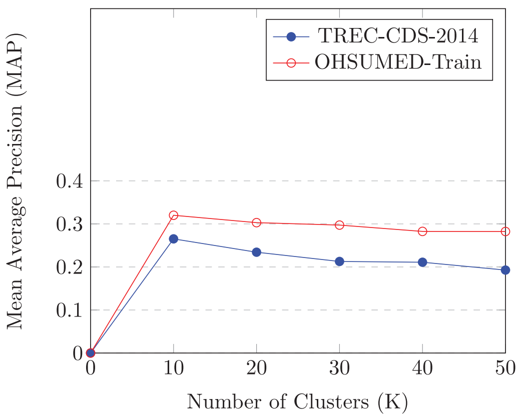

Fig. 7. Quality of documents returned at different cluster number for TREC-CDS-2014 and OHSUMED-train dataset.

- *Binary Preference (bpref):* This measure depends on the number of relevant documents that have been retrieved. Like other measures, it does not consider the rank of retrieved documents, rather it depends only on the relative ranks of judged documents. It is calculated as follows:

$$bpref = \frac{1}{R}\sum_{r} 1 - \frac{n_r}{R} \tag{20}$$

Here, n_r is the rank of first irrelevant retrieved document amongst judged documents and higher than r, which is relevant and retrieved document. R is the total number of judged documents. For example when *bpref*= 70% then 30% irrelevant documents are retrieved before relevant document. Both *bpref* and *MAP* are highly correlated when used with entire results. The *bpref* gives the correct ranking for a single query and incomplete judgments, whereas *MAP* does not.

4.2.3. Parameter selection

In order to improve the quality of the results using fine-tuned parameters, we conducted several experiments on the TREC-CDS dataset using the 2014-track query set, the OHSUMED-Train dataset, and the NPL, LISA, CACM dataset, thus, identified the most appropriate parameters for each algorithm. We tested the performance of the system using these parameters against the TREC-CDS dataset using the 2015-track query set and the OHSUMED-Test dataset. In other words, the aim of these experiments is to fix the cluster number (K) for the K-Flock algorithm, and the minimum support count value (s_{min}) for the RElim algorithm. To deal with the varying cluster size of large datasets, we considered the relative minimum support count for a very large dataset, as discussed in Section 3.3. In Algorithm 1, the value of variable $MAX-ITERATIONS$ is fixed to 10 just to ensure an infinity condition, because the number of iterations are dynamically determined during runtime, based on clusters formed in two successive iterations.

Cluster number (K) is determined by considering the silhouette coefficient index. It can be seen from Tables 4 and 5; for TREC-CDS 2014, OHSUMED-Train and CACM datasets, the silhouette coefficient index generated with K = 10 has higher values than all other K values, which simply indicates that 10 clusters formed are dense and well separated from each other. Similarly for NPL and LISA datasets, silhouette index generated with K = 5 has higher index values than other K values. Although we used the silhouette coefficient index to decide K values but to double-check it with the proposed system, we performed experiments with all K values.

The quality of the returned documents is measured in terms of mean average precision and is shown in Figs. 6 and 7. Cluster number varies from 5 to 30 in the interval of 5 for NPL, LISA, and CACM datasets; and 10 to 50 in the interval of 10 for the TREC-CDS 2014 and the OHSUMED-Train dataset.

Based on the above preliminary experiments, the values of the parameters used for the test datasets are shown in Table 6.

4.3. Experimental results

As discussed in Section 4.3, we conducted several experiments to tune parameters and presented the results with the best combination on five varying size datasets. Our experiments mainly investigate the effect of the K-Flock clustering technique combined with RElim on document retrieval using a probabilistic retrieval model. As the K-Flock clustering algorithm is improved over K-means clustering, we have therefore conducted exhaustive experiments to compare the results with the K-means IR approach and the Sequential IR approach. The quality of retrieval documents is evaluated across different IR measures defined in Section 4.2; we also recorded the average retrieval taken by each model.

Table 7 presents the experimental results in terms of MAP, P@5, P@10, bpref, and Rprec measures. The values generated by the K-Flock algorithm are shown in bold. We also validated the results on two test datasets, as shown in Table 8. The results confirm that our proposed framework significantly outperforms the K-means based approach and the sequential IR approach. Tables 7 and 8 also show that the average retrieval time taken by each query to retrieve the top 10 documents is more for K-Flock than the K-means algorithm, due to the fuzzy nature of clusters; however, it is less w.r.t. The Sequential IR approach. Fig. 8 shows the graphical comparisons of average retrieval time taken for each dataset. It is also inferred that larger datasets take longer retrieval time compared to small datasets.

4.4. Comparative evaluation with state-of-the-art IR frameworks

In this section, we presented the performance comparison between the proposed IR framework and the state-of-the-art IR approaches. We compared each approach on the Mean Average Precision (MAP) value of the IR measure. The values used for comparison are the best reported values by each of the approaches. Since we have not found common approaches for all datasets, we compared them separately with the available studies. Tables 9, 10, and 11 shows the results for various IR approaches. The bold values in the tables indicate the best value of the MAP generated by the specific approach.

Table 6
Parameter Settings required for IR Framework.

Parameter	Description	TREC-CDS & OHSUMED Dataset	NPL & LISA Dataset	CACM Dataset
K	Number of clusters required	10	5	10
s_{min}	Relative minimum support count	Relative Support count as defined in Eq. (7)	50%	50%
$MAX-ITERATION$	Maximum number of iterations allowed if clustering algorithm does not converge	10	10	10

Table 7
Quality of documents returned using tuned parameter over Training Datasets (ART/Q @10 denotes the average retrieval time in seconds taken by a query to retrieve top 10 documents; Seq-IR denotes Sequential IR approach).

Dataset	Model	P@5	P@10	MAP	bpref	Rprec	ART/Q @10 (sec) (sec)
TREC-CDS 2014	K-Flock IR	**0.4400**	**0.4533**	**0.2652**	**0.5292**	**0.4984**	**0.7230**
	K-means IR	0.3400	0.3800	0.2230	0.5029	0.4493	0.7190
	Seq-IR	0.4200	0.4133	0.2097	0.4820	0.4317	0.8920
OHSUMED- Train	K-Flock IR	**0.4286**	**0.4190**	**0.3200**	**0.5821**	**0.4329**	**0.2420**
	K-means IR	0.4032	0.3794	0.2913	0.5423	0.3805	0.2200
	Seq-IR	0.3937	0.3730	0.2768	0.5126	0.3800	0.2720
NPL	K-Flock IR	**0.3720**	**0.3538**	**0.2259**	**0.4193**	**0.3814**	**0.1600**
	K-means IR	0.2688	0.2720	0.1584	0.3426	0.3102	0.1530
	Seq-IR	0.2839	0.2731	0.1288	0.3023	0.2664	0.2150
LISA	K-Flock IR	**0.3600**	**0.3143**	**0.2695**	**0.4592**	**0.3907**	**0.1630**
	K-means IR	0.3029	0.2800	0.2057	0.4113	0.3479	0.1560
	Seq-IR	0.1943	0.1486	0.1622	0.2667	0.2517	0.1890
CACM	K-Flock IR	**0.4654**	**0.3885**	**0.3536**	**0.4907**	**0.4781**	**0.1230**
	K-means IR	0.4269	0.3808	0.2813	0.4262	0.4113	0.1210
	Seq-IR	0.3808	0.3154	0.2154	0.3696	0.3476	0.1800

Table 8
Quality of documents returned using tuned parameter over Testing Datasets (ART/Q @10 denotes the average retrieval time in seconds taken by a query to retrieve top 10 documents; Seq-IR denotes Sequential IR approach).

Dataset	Model	P@5	P@10	MAP	bpref	Rprec	ART/Q @10
TREC-CDS 2015	K-Flock IR	**0.5267**	**0.5533**	**0.2841**	**0.5245**	**0.5220**	**0.6980**
	K-means IR	0.5200	0.5333	0.2676	0.5136	0.5068	0.6890
	Seq-IR	0.4467	0.4500	0.1915	0.4358	0.4235	0.7420
OHSUMED- Test	K-Flock IR	**0.4571**	**0.5032**	**0.3026**	**0.5575**	**0.4812**	**0.4500**
	K-means IR	0.4159	0.4317	0.2705	0.5524	0.4381	0.4340
	Seq-IR	0.3810	0.4000	0.2359	0.4950	0.4005	0.5180

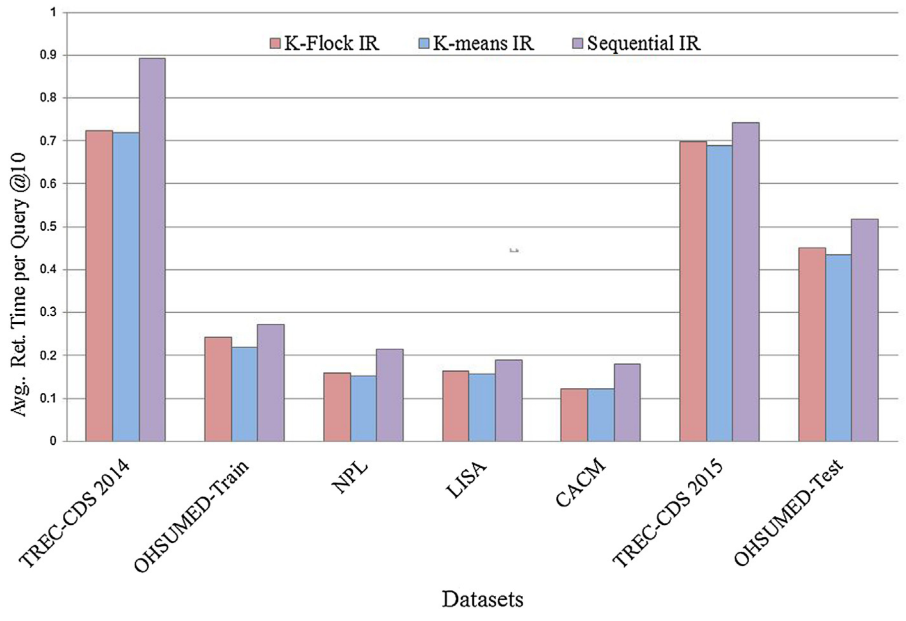

Fig. 8. Average Retrieval Time (ART) taken by each query to retrieve top 10 documents.

Table 9
Comparison of Mean Average Precision (MAP) value generated by the proposed approach and other existing approaches on CDS-2014 and CDS 2015 Datasets.

Approach	CDS 2014	CDS 2015
Document-based Neural Relevance Model (DNRM) (Ran, He, Hui, Xu, & Sun, 2017)	0.1578	0.1878
PRF + Embedding based technique (Yang et al., 2017)	0.1661	0.1806
Self-adaptive SDF for Probabilistic IR (Marchesin et al., 2019)	0.1883	0.1807
NLP-based feature weighting with classification and clustering (Song, Hu, He, & Dou, 2019)	**0.2807**	0.2302
K-means IR	0.2230	0.2676
K-Flock IR	0.2652	**0.2841**

Table 10
Comparison of Mean Average Precision (MAP) value generated by the proposed approach and other existing approaches on OHSUMED Dataset.

Approach	OHSUMED
Firefly based PRF (Khennak & Drias, 2016)	0.1540
APSO based QE (Khennak & Drias, 2017)	0.1840
Fuzzy K-means & Firefly + APSO based QE (Khennak & Drias, 2018)	0.1840
K-medoids cluster-based Query Expansion (Khennak, Drias, Kechid, & Moulai, 2019)	0.1850
PSO-based BM25 IR (Thakur, Mehrotra, Bansal, & Bala, 2019)	0.1900
Deep Relevance Matching Model DRMM (Marchesin et al., 2019)	0.2720
Neural Vector Space Model NVSM (Marchesin et al., 2019)	0.2140
K-means IR	0.2705
K-Flock IR	**0.3026**

Table 11
Comparison of Mean Average Precision (MAP) value generated by the proposed approach and other existing approaches on NPL, LISA, CACM Datasets. (*na* indicates value is not available in the referred paper).

Approach	NPL	LISA	CACM
Essential LSI model (Kontostathis, 2007)	0.1420	**0.2770**	0.2290
Fuzzy PSO + semantic filtering based QE (Gupta & Saini, 2017)	na	na	0.2852
Nature-inspired cuckoo search & Fuzzy APSO based QE (Sharma et al., 2019a)	na	na	0.2316
Fuzzy APSO based QE (Sharma, Pamula, & Chauhan, 2019)	na	na	0.2791
Distributional Semantic based QE (da Silva & Maia, 2019)	**0.2282**	0.2602	na
K-means IR	0.1584	0.2057	0.2813
K-Flock IR	0.2259	0.2695	**0.3536**

It can be seen from the Table 9, an NLP based feature weighting model with the application of classification and clustering method studied in Sankhavara (2020), which produced competitive results compared to our work. However, the model needs a feedback query relevant document; therefore, human intervention is required. For the OHSUMED dataset in Table 10, the proposed model outperformed over all other approaches. It can be seen from Table 11 that the proposed model produced competitive results w.r.t. other approaches. In general, it is observed that some embedding based query expansion models have shown competitive results against the proposed model. However, the embedding based techniques require pre-trained models, and the expanded queries usually take more time to generate results. On the contrary, in the proposed work, only query frequent patterns are required to be generated when a query arrives; therefore, we argue that our model takes less time to execute. It is also inferred from the comparison that the proposed bio-inspired K-Flock algorithm, when integrated with RElim, has produced better results compared to separately used bio-inspired or data mining IR approaches.

5. Conclusion and future scope

A significant contribution of our work is, the proposed K-Flock document clustering algorithm inspired by the heuristic behaviors of boids in the search space. The movement of multiple search agents is governed by four rules, and mimicking of swarming behavior allows these agents to cooperatively steer towards an optimal objective within a reasonable time. As a result, the problem of falling into local minima gets resolved. Moreover, the algorithm allowed boids to have a membership for more than one cluster; the fuzzy characteristic of the cluster helps to measure the relevance of each document without missing a single instance.

We used a memory-efficient RElim FPM algorithm with varying relative support counts for different volume data. FPM techniques help to gain more insight through the context of the large dataset in less time. The RElim algorithm is used to derive closed frequent patterns from the fuzzy clusters. Later, we suggested a probabilistic document retrieval technique to assign the probability values of each cluster based on the matching closed patterns between clusters and the queries. These matching scores are used to determine the number of query relevant documents to be retrieved from each cluster.

To assess the performance of the proposed framework, we have extensively carried out experiments on five well-known variable size datasets and evaluated performance on two test datasets. We also compared the performance of the proposed framework with recent IR studies and state-of-the-art IR approaches. In particular, the performance of the proposed approach has significantly improved the quality of retrieved documents over other IR techniques. The results are consistent with our hypothesis that swarm intelligence techniques, when integrated with the mining techniques, could enhance the ability of the IR model to improve retrieval performance.

In the future, we are planning to combine neural network techniques with the swarm intelligence technique for document information retrieval tasks. This work does not include the semantic relatedness between closed frequent patterns of the clusters and the query, although this may affect the final retrieval results. In recent years, significant work has been performed in the field of word representation. Advanced word embedding techniques can be helpful in understanding the semantic relationship between the query and the clusters. Therefore, it will be interesting to explore these things in future research work.

Declaration of Competing Interest

There is no Conflict of Interest.

Credit authorship contribution statement

Amol P. Bhopale: Conceptualization, Methodology, Software, Validation, Investigation, Data curation, Writing - original draft, Visualization. **Ashish Tiwari:** Conceptualization, Methodology, Supervision.

References

Abboud, Y., Brun, A., & Boyer, A. (2019). C3ro: An efficient mining algorithm of extended-closed contiguous robust sequential patterns in noisy data. *Expert Systems with Applications, 131*, 172–189.

Babashzadeh, A., Daoud, M., & Huang, J. (2013). Using semantic-based association rule mining for improving clinical text retrieval. In *Proceedings of the international conference on health information science* (pp. 186–197). Berlin, Heidelberg: Springer.

Beil, F., Ester, M., & Xu, X. (2002). Frequent term-based text clustering. *Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 436–442). Edmonton, Alberta, Canada: ACM.

Biswas, P., & Pal, B. B. (2019). A fuzzy goal programming method to solve congestion management problem using genetic algorithm. *Decision Making: Applications in Management and Engineering, 2*(2), 36–53.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research, 3*(Jan), 993–1022.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm intelligence: From natural to artificial systems*. New York, NY: Oxford university press.

Borgelt, C. (2005). Keeping things simple: finding frequent item sets by recursive elimination. In *Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations* (pp. 66–70). Chicago, Illinois, USA: ACM.

Buscher, G., Dengel, A., Biedert, R., & Elst, L. V. (2012). Attentive documents: Eye tracking as implicit feedback for information retrieval and beyond. *ACM Transactions on Interactive Intelligent Systems (TiiS), 1*(2), 9.

Chawla, S. (2016). A novel approach of cluster based optimal ranking of clicked urls using genetic algorithm for effective personalized web search. *Applied Soft Computing, 46*, 90–103.

Collett, T. S., Graham, P., Harris, R. A., & Hempel-de Ibarra, N. (2006). Navigational memories in ants and bees: Memory retrieval when selecting and following routes. *Advances in the Study of Behavior, 36*, 123–172.

Croft, W. B., & Harper, D. J. (1979). Using probabilistic models of document retrieval without relevance information. *Journal of Documentation, 35*(4), 285–295.

Croft, W. B., Metzler, D., & Strohman, T. (2010). *Search engines: Information retrieval in practice*: 520. USA: Addison-Wesley Reading.

Cui, X., Gao, J., & Potok, T. E. (2006). A flocking based algorithm for document clustering analysis. *Journal of Systems Architecture, 52*(8–9), 505–515.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science, 41*(6), 391–407.

Djenouri, Y., Belhadi, A., & Belkebir, R. (2018). Bees swarm optimization guided by data mining techniques for document information retrieval. *Expert Systems with Applications, 94*, 126–136.

El Mahdaouy, A., El Alaoui, S. O., & Gaussier, E. (2018). Improving arabic information retrieval using word embedding similarities. *International Journal of Speech Technology, 21*(1), 121–136.

Fan, W., Gordon, M. D., & Pathak, P. (2004). Discovery of context-specific ranking functions for effective information retrieval using genetic programming. *IEEE Transactions on Knowledge and Data Engineering, 16*(4), 523–527.

Fournier-Viger, P., Lin, J. C.-W., Nkambou, R., Vo, B., & Tseng, V. S. (2019). *High-utility pattern mining*. Springer.

Fung, B. C., Wang, K., & Ester, M. (2003). Hierarchical document clustering using frequent itemsets. In *Proceedings of the 2003 SIAM international conference on data mining* (pp. 59–70). Philadelphia: SIAM.

Gan, W., Lin, J. C.-W., Fournier-Viger, P., Chao, H.-C., Hong, T.-P., & Fujita, H. (2018). A survey of incremental high-utility itemset mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8*(2), e1242.

Ganguly, S. (2020). Multi-objective distributed generation penetration planning with load model using particle swarm optimization. *Decision Making: Applications in Management and Engineering, 3*(1–2), 30–42.

Gupta, Y., & Saini, A. (2017). A novel fuzzy-pso term weighting automatic query expansion approach using combined semantic filtering. *Knowledge-based Systems, 136*, 97–120.

Han, X., Qiang, Y., & Lan, Y. (2017). A bird flock gravitational search algorithm based on the collective response of birds. *The Computer Journal, 60*(11), 1687–1716.

Jin, X., Agun, D., Yang, T., Wu, Q., Shen, Y., & Zhao, S. (2016). Hybrid indexing for versioned document search with cluster-based retrieval. In *Proceedings of the 25th acm international on conference on information and knowledge management* (pp. 377–386). Indianapolis, IN, USA: ACM.

Jung, J. J. (2012). Evolutionary approach for semantic-based query sampling in large-scale information sources. *Information Sciences, 182*(1), 30–39.

Khennak, I., & Drias, H. (2016). A firefly algorithm-based approach for pseudo-relevance feedback: Application to medical database. *Journal of medical systems, 40*(11), 240.

Khennak, I., & Drias, H. (2017). An accelerated pso for query expansion in web information retrieval: Application to medical dataset. *Applied Intelligence, 47*(3), 793–808.

Khennak, I., & Drias, H. (2018). Data mining techniques and nature-inspired algorithms for query expansion. In *Proceedings of the international conference on learning and optimization algorithms: Theory and applications* (pp. 1–6).

Khennak, I., Drias, H., Kechid, A., & Moulai, H. (2019). Clustering algorithms for query expansion based information retrieval. In *International conference on computational collective intelligence* (pp. 261–272). Cham: Springer.

Kontostathis, A. (2007). Essential dimensions of latent semantic indexing (lsi). *2007 40th annual hawaii international conference on system sciences (hicss'07)*. IEEE. 73–73

Lafferty, J., & Zhai, C. (2001). Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th annual international acm sigir conference on research and development in information retrieval* (pp. 111–119). New Orleans, Louisiana, USA: ACM.

Landauer, T. K., Foltz, P. W., & Laham, D. (1998). An introduction to latent semantic analysis. *Discourse processes, 25*(2–3), 259–284.

Lin, C.-H., Chen, H.-Y., & Wu, Y.-S. (2014). Study of image retrieval and classification based on adaptive features using genetic algorithm feature selection. *Expert Systems with Applications, 41*(15), 6611–6621.

Loper, E., & Bird, S. (2002). Nltk: The natural language toolkit. *arXiv preprint cs/0205028*.

Marchesin, S., Purpura, A., & Silvello, G. (2019). Focal elements of neural information retrieval models. an outlook through a reproducibility study. *Information Processing & Management*, 102109.

Mei, J.-P., & Chen, L. (2014). Proximity-based k-partitions clustering with ranking for document categorization and analysis. *Expert Systems with Applications, 41*(16), 7095–7105.

Naini, K. D., Altingovde, I. S., & Siberski, W. (2016). Scalable and efficient web search result diversification. *ACM Transactions on the Web (TWEB), 10*(3), 15.

Picard, D., Revel, A., & Cord, M. (2012). An application of swarm intelligence to distributed image retrieval. *Information Sciences, 192*, 71–81.

Ponte, J. M., & Croft, W. B. (1998). *A language modeling approach to information retrieval*. University of Massachusetts at Amherst Ph.D. thesis..

Ran, Y., He, B., Hui, K., Xu, J., & Sun, L. (2017). A document-based neural relevance model for effective clinical decision support. In *2017 ieee international conference on bioinformatics and biomedicine (BIBM)* (pp. 798–804). Kansas City, MO: IEEE.

Rawashdeh, M., & Ralescu, A. (2012). Crisp and fuzzy cluster validity: Generalized intra-inter silhouette index. In *2012 annual meeting of the north american fuzzy information processing society (NAFIPS)* (pp. 1–6). Berkeley, CA: IEEE.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *Acm siggraph computer graphics: vol. 21.* (21, pp. 25–34). New York, NY, United States: ACM.

Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M. M., Gatford, M., et al. (1995). Okapi at trec-3. *Nist Special Publication Sp, 109*, 109.

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing & Management, 24*(5), 513–523.

Salton, G., & McGill, M. J. (1983). Introduction to modern information retrieval. *McGraw-Hill computer science series*. USA: McGraw-Hill.

Sankhavara, J. (2020). Feature weighting in finding feedback documents for query expansion in biomedical document retrieval. *SN Computer Science, 1*(2), 1–7.

Sharma, D. K., Pamula, R., & Chauhan, D. (2019). A hybrid evolutionary algorithm based automatic query expansion for enhancing document retrieval system. *Journal of Ambient Intelligence and humanized computing*, 1–20.

da Silva, F. T., & Maia, J. E. (2019). Query expansion in text information retrieval with local context and distributional model. *Journal of Digital Information Management, 17*(6), 313.

Sharma, D. K., Pamula, R., & Chauhan, D. (2019). Soft computing techniques based automatic query expansion approach for improving document retrieval. In *2019 amity international conference on artificial intelligence (AICAI)* (pp. 972–976). Dubai, United Arab Emirates: IEEE.

Song, Y., Hu, W., He, L., & Dou, L. (2019). Enhancing the healthcare retrieval with a self-adaptive saturated density function. *Pacific-asia conference on knowledge discovery and data mining* (pp. 501–513). Cham: Springer.

Thakur, N., Mehrotra, D., Bansal, A., & Bala, M. (2019). A novel multi-parameter tuned optimizer for information retrieval based on particle swarm optimization. *International Journal of Recent Technology and Engineering, 8*(3), 1723–1731.

Van, T., Vo, B., & Le, B. (2018). Mining sequential patterns with itemset constraints. *Knowledge and Information Systems, 57*(2), 311–330.

Wei, X., & Croft, W. B. (2006). LDA-based document models for ad-hoc retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 178–185). Seattle, Washington, USA: ACM.

Wiemer-Hastings, P., Wiemer-Hastings, K., & Graesser, A. (2004). Latent semantic analysis. In *Proceedings of the 16th international joint conference on artificial intelligence* (pp. 1–14). San Francisco, California, US: Citeseer.

Yang, C., He, B., Li, C., & Xu, J. (2017). A feedback-based approach to utilizing embeddings for clinical decision support. *Data Science and Engineering, 2*(4), 316–327.

Yu, H., Searsmith, D., Li, X., & Han, J. (2004). Scalable construction of topic directory with nonparametric closed termset mining. In *Fourth IEEE international conference on data mining (icdm'04)* (pp. 563–566). Brighton, UK: IEEE.

Zhang, L., Mei, T., Liu, Y., Tao, D., & Zhou, H.-Q. (2011). Visual search reranking via adaptive particle swarm optimization. *Pattern Recognition, 44*(8), 1811–1820.

Zhong, N., Li, Y., & Wu, S.-T. (2012). Effective pattern discovery for text mining. *IEEE Transactions on Knowledge and Data Engineering, 24*(1), 30–44.