N° d'ordre : 12/2017-C/INF

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université des Sciences et de la Technologie Houari Boumediène

Faculté d'Electronique et d'Informatique

جامعة هواري بومدين
للعلوم والتكنولوجيا
U S T H B

THESE

Présentée pour l'obtention du **diplôme** de **DOCTORAT 3eme Cycle (LMD)**

En : Informatique

Spécialité : Intelligence Artificielle

Par : KHENNAK Ilyes

Sujet

Intelligence en Essaim et Optimisation Combinatoire pour les Moteurs de Recherche Web

Soutenue publiquement, le 23/04/2017, devant le jury composé de :

M DJOUADI Yassine	Professeur à l'USTHB	Président
Mme DRIAS Habiba	Professeur à l'USTHB	Directrice de thèse
M BOUGHANEM Mohand	Professeur à l'Univ Toulouse 3	Examinateur
M AHMED-OUAMER Rachid	Professeur à l'UMMTO	Examinateur
M BOUROUBI Sadeg	Professeur à l'USTHB	Examinateur
Melle BOUGHACI Dalila	Professeur à l'USTHB	Examinatrice
M KECHID Samir	Professeur à l'USTHB	Invité

UNIVERSITY OF SCIENCES AND TECHNOLOGY
HOUARI BOUMEDIENE

DOCTORAL THESIS

Swarm Intelligence and Combinatorial Optimization for Web Search Engines

Author:
Ilyes KHENNAK

Supervisor:
Prof. Habiba DRIAS

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Computer Science

in the

Laboratory of Research in Artificial Intelligence
Department of Computer Science

Jury's members:

Djouadi Yassine	Professor, USTHB	President
Drias Habiba	Professor, USTHB	Thesis director
Boughanem Mohand	Professor, Univ Toulouse 3	Examiner
Ahmed-Ouamer Rachid	Professor, UMMTO	Examiner
Bouroubi Sadeg	Professor, USTHB	Examiner
Boughaci Dalila	Professor, USTHB	Examiner
Kechid Samir	Professor, USTHB	Invited

UNIVERSITY OF SCIENCES AND TECHNOLOGY HOUARI BOUMEDIENE

Abstract

Faculty of Electronic and Computer Science
Department of Computer Science

Doctor of Computer Science

Swarm Intelligence and Combinatorial Optimization for Web Search Engines

by Ilyes KHENNAK

Swarm Intelligence (SI) algorithms are now among the most widely used soft computing techniques for optimization and computational intelligence. Due to their sufficient simplicity and high flexibility, many recent SI algorithms have begun to receive more attention in the literature, such as: Accelerated Particle Swarm Optimization (APSO), Firefly Algorithm (FA) and Bat Algorithm (BA). In this work, we propose the application of SI to efficiently solve the problem of Query Expansion (QE) in Web Information Retrieval (IR). Unlike prior efforts, we introduce a novel modelling of QE that aims to find the suitable expanded query from among a set of expanded query candidates. However, due to the huge number of potential expanded query candidates, it is extremely complex to produce the best one through conventional hard computing methods. Therefore, we propose to consider the problem of QE as a combinatorial optimization problem and use SI to solve this issue. We evaluate the proposed SI for QE on MEDLINE, the world Web's largest medical library. We conduct a preliminary experiment to tune the SI parameters. Then, we compare the results to the state-of-the-art methods. The experimental analysis demonstrates that the proposed SI for QE is very competitive and yields a substantial improvement over the other methods in terms of retrieval effectiveness.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The data explosion

Nowadays, the volume of information available on the World Wide Web is constantly increasing and the number of published websites is continuously growing. For instance, the total number of websites had risen from 200 million in 2010 to 900 million in 2014 and this number is expected to exceed 1 Billion in 2016. In addition, the amount of user-generated content posted on websites is exponentially expanding, especially on social networking sites. Every minute, Facebook and Twitter users share nearly 2.5 million pieces of content and 300 thousand tweets, respectively. The number of search queries has also considerably grown. In 2012, Google received 2 million queries per minute and this number had doubled in 2014. Besides the rapidly growing amount of data, the Internet traffic has dramatically increased. According to the latest Cisco study, the Internet data traffic will reach 1 ZB by 2016, and it is expected to double in 2019. This is mainly due to the large number of devices connected to the net, such as: PCs, smartphones, tablets, and machine-to-machine (M2M).

1.2 The challenges of Web IR

The unprecedented explosion of information that the Web is experiencing has led to the following results:

- New words are continuously being introduced in the World Wide Web. According to Williams and Zobel [78], there is one new word in every

two hundred words. Studies by [78][16][73] showed that this invasion is mainly owing to: neologisms, first occurrences of rare personal and place names, abbreviations, acronyms, emoticons, URLs and typographical errors.

- The Web users are constantly exploiting these new words in their search queries. Chen et al.[14] indicated in their study that more than 17% of query keywords are out of vocabulary, 45% of them are E-speak (lol), 18% are companies and products such as new medicament or viruses names, 16% are proper names, 15% are misspellings and foreign words [72][1].

Undoubtedly, the difficulty of disambiguating the senses of these imprecise and unclear keywords will certainly cause the failure of Web search engines to find the desired information.

1.3 Query expansion in Web IR

One of the most powerful and effective method that may overcome the above drawback is Query Expansion (QE). This method aims to augment the original search queries with additional keywords that best characterize the users' needs [12]. QE is currently used in various applications such as multimedia information retrieval [48], question answering [58], information filtering [44], sport and game [2], medical and health [57][15] and e-commerce [66].

In the last years, owing to the dramatic growth of the amount of data available on the Internet, an enormous number of QE techniques have been proposed to enhance the quality of Web information retrieval results. They employed a variety of methods that depend on several data sources and use advanced approaches to find the best expansion keywords. However, these techniques failed to improve the retrieval performance of Web search engines and returned irrelevant information. This is mainly due to the fact that all of these techniques were based on traditional modelling of query expansion, which is basically designed to look for the best expansion keywords and not really the

appropriate expanded query. As a result, these proposed QE techniques could not be regularly employed in the major operational Web IR systems [12].

1.4 Hard computing for QE in Web IR

To cope with this limitation and solve the problem of query expansion in Web IR systems, we propose a new modelling of QE that aims to find the suitable expanded query instead of generating the best expansion keywords. In this new modelling, the best expanded query is selected from among a set of expanded query candidates. This set of expanded query candidates includes all possible combinations of keywords belonging to the pseudo-relevant documents. Due to the huge number of potential expanded query candidates, it is too complex to find the best one through conventional hard computing methods as they require incredibly large processing time, especially for large-scale data sources such as the Web.

1.5 Swarm intelligence for QE in Web IR

To overcome this drawback and find the appropriate expanded query within reasonable time, we consider the problem of query expansion as a combinatorial optimization problem and apply swarm intelligence algorithms to solve this issue. Swarm intelligence algorithms are soft computing techniques inspired by nature and mimicking some effective characteristics of animal swarms [84]. These algorithms are efficient in solving difficult combinatorial optimization problems. In swarm intelligence algorithms, we sacrifice the guarantee of finding optimal solutions for the benefit of getting good solutions in a considerably shorter amount of time. Good examples of swarm algorithms are Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO). One of the recent swarm algorithms that has a great potential success is Bat Algorithm (BA) proposed by Yang in 2010 [83]. BA is based on the simulation of the echolocation phenomenon of bats. The interest exhibited by the scientists for this algorithm, still increases day by day in the literature because of its

flexible structure and clear and simple modelling. Moreover, it includes important features brought from previous evolutionary approaches such as a local search, a random walk strategy, a simulated annealing convergence strategy and movement rules of particle swarm optimization. Firefly Algorithm (FA) is another recent best swarm intelligence algorithm, which was developed by Yang in 2008 [79][80]. It is based on the flashing behavior of fireflies and the phenomenon of bioluminescent communication. Accelerated Particle Swarm Optimization (APSO) was also introduced by Yang in 2008 [81]. It is a simplified version of particle swarm optimization (PSO) which has global search ability, fast convergence speed and strong robustness.

1.6 Research goals

In this thesis, we involve the three above SI algorithms (i.e., BA, FA and APSO) to solve the problem of query expansion in Web information retrieval and produce the best expanded query within reasonable time. The overall procedure of SI for QE consists in, first selecting the pseudo-documents to the initial query, extract their keywords to constitute potential solutions and finally launch the SI to determine the best expanded query. We extensively evaluate the proposed SI for QE on MEDLINE, the world's largest medical library. We first conduct a preliminary experiment to fix the parameters of BA, FA and APSO. Then, we compare the results to the state-of-the-art methods. We also compare the results with recently published methods for query expansion which involved swarm intelligence algorithms and evolutionary algorithms.

1.7 Thesis organization

The rest of this thesis is organized as follows:

- Chapter **2** provides an overview about information retrieval and query expansion. It briefly reviews IR basis and query expansion techniques.

- Chapter **3** surveys swarm intelligence algorithms, covering bat algorithm, firefly algorithm and accelerated particle swarm optimization.

- Chapter **4** introduces our proposed approach to solve the problem of query expansion using swarm intelligence algorithms.

- Chapter **5** describes the simulation results of different approaches and their comparison.

- Chapter **6** presents the conclusion about the work done in this thesis.

Chapter 2

IR and Query Expansion

The massive influx of new features on the web such as first occurrences of proper names, abbreviations, misspelling words, etc., as well as the use of these ambiguous and imprecise features to describe the user need have resulted in a failure to retrieve the relevant information. Furthermore, the term mismatch problem or vocabulary problem still remains one of the most obstacles presently facing the retrieval effectiveness. To cope with these critical issues, several methods have been suggested including interactive query refinement and word sense disambiguation. However, expanding the original query with additional keywords is one of the most successful and promising techniques to enhance the retrieval effectiveness of document ranking.

In this chapter, we provide a brief introduction to query expansion. We first discuss IR on the Web (Section 2.1). Then, we describe the process of information retrieval (Section 2.2). In Section 2.3, we present query expansion techniques and some recent works that involved query expansion in Web information retrieval.

2.1 IR on the Web

Information Retrieval on the web has become an important area of research in computer science. The aim is to satisfy the information needs of users.

Many web-based information retrieval tools have been proposed and developed to provide users with relevant and accurate information. The directory and the search engine are two well-known search tools: (i) a *directory* is

an index of sites, which is categorized according to a tree structure and referenced by descriptive records. In general, the search is performed by traversing the tree. The directory is especially useful for exploring a general topic or finding similar sites on the same topic. (ii) A *search engine* is a search tool for finding resources on the Internet. These resources are automatically identified and indexed through intelligent agents. In general, search engine retrieves and classifies resources in relation to the keywords provided by users [35].

Information retrieval on the web uses the same indexing and matching techniques as the standard IR, but with new challenges. These challenges are summarized as follows: (i) *Web is huge*, the volume of information available on the Web is constantly increasing and the number of published websites is continuously growing. For instance, the total number of websites had risen from 200 million in 2010 to 900 million in 2014 and this number exceeded 1 Billion in 2016. Accordingly, Information retrieval on the web will become more challenging to retrieve relevance resources. (ii) *Web is dynamic*, information and resources on the Web can be created, updated, and deleted continuously. This dynamic nature of the Web makes the update of indexes highly difficult. (iii) *Web is duplicated*, about 50% of the Web resources are duplicated or replicated. Web search tools try to avoid indexing these duplicated resources so as not to affect search results. (iv) *Web is unreliable*, various sources of information add and update resources on the Web. These sources may be reliable or not. Thus, the quality of resources found can be good or bad. (v) *Web and Spammers*, Spammers benefit from the advantages of hyperlinks and manipulate automatic search algorithms to redirect results and increase traffic to their sites [24][47].

Information retrieval on the Web also differs from standard IR in terms of how users search for information: (i) One way is to find particular websites through direct entry, or navigating links between websites. (ii) Another way is to find websites of interactions and transactions to make purchases, download files, launch applications, etc [4].

2.1.1 Web search engines

A web search engine is an information retrieval system designed to find resources on the World Wide Web, such as web pages, images, videos, etc. In other words, it is a search tool that allows users to easily access data and applications over the Internet. Three well-known examples of web search engines are Google, Yahoo, and Bing.

Usually, the input of Web search engine is formed by a set of keywords entered by users; whereas, the output is a list of all the resources related to those keywords, sorted according to their degree of relevance. This is a common feature of all search engines. Other common features include: (i) the collection of resources, (ii) indexing and storing the collected resources, and finally (iii) searching for any collected resource containing information in relation to the keywords entered by the Internet users. As for the indexing process, search engines use intelligent agents, called Robots, to find new resources, these represent resources, and finally index them automatically [46].

The standard indexing of Web pages does not take into account the meaning of the keywords describing these resources. Therefore, web search engines may not know whether the results returned are relevant or not. In order to overcome this shortcoming, Semantic Web was introduced as an extension of the Web. This extension allows the search engines to directly exploit the semantics of resources and understand the content of data before returning the information desired by the Internet users.

2.1.2 Semantic Web

As mentioned above, the amount of information and resources available on the Web is constantly increasing. This increase has revealed several limitations and drawbacks. Indeed, Web search engines, the ones that were developed at the beginning of the Internet, did not have the tools to represent and structure the resources satisfactorily. Adding these tools to web search engines will certainly prevent the large number of results returned to users and allow direct access to relevant resources.

Accordingly, the Semantic Web, which is an extension of the Web search engines, has been introduced to reduce irrelevant resources and increase the usefulness of the Web. This extension takes into account the meanings of the resources in order to exploit their semantics and cooperate with the users to find only the relevant resources. In other words, Semantic Web aims to be a bridge between machine and human reasoning [25].

The Semantic Web was mainly designed to bypass the disadvantages of the Web, such as: (i) the absence of semantics of hypertext links, which make them unused by Web search engines. (ii) The metadata used are not structured. (iii) Reasoning about the knowledge described in the web pages cannot be done because there is no conceptual resources allowing the semantic representation of this knowledge. Several solutions have been proposed by the Semantic Web to overcome those drawbacks. These solutions include: (i) proposals for formalisms to represent the semantic content of resources, (ii) proposals for conceptual resources for modeling knowledge, and (iii) proposals for languages defined for all metadata.

2.2 IR basis

Information retrieval (IR) is a research area that focuses on the representation, storage, organization and access to information. The main objective of IR is to find, among the large volume of available resources, those that satisfy and meet the user's needs. In other words, information retrieval is the set of methods and techniques that facilitate accessing relevant information to a user with an information need [53].

The information retrieval process is carried out and realized through an information retrieval system (IRS). The purpose of this system is to implement techniques and mechanisms to retrieve and select relevant documents in response to an information need expressed by a user.

The information retrieval system includes two main functionalities: (i) *Indexing*: which consists in representing the documents by significant elements.

(ii) *Matching*: which consists of comparing the representations of the queries with those of the documents.

The information retrieval system brings out constitutive notions and concepts, such as: (i) *Document*: a resource that the user may want to retrieve. This resource can be a text file, a text fragment, a web page, an image, a video, etc. It can be structured, semi-structured or unstructured. (ii) *Corpus* (or collection of documents): the set of documents, which are understandable and accessible, in which a user wishes to obtain information. (iii) *Query* (or Information need): the set of keywords, terms, or expressions characterizing the information that the user is looking for. (iv) *Relevance*: a measure that indicates whether a document satisfies the user's need. This measurement is a fundamental notion in the evaluation of the performance of IRS. (v) *Similarity*: a measure that indicates if the document matches the query.

2.2.1 IR process

In the following, we present the fundamental features of an information retrieval system, namely indexing and matching.

a. Indexing

Indexing (representation of information) is a process of representing the content of each document in the collection and the content of the query by a set of keywords or terms. The main purpose of this indexing phase is to facilitate the comparison between the representation of the document and the representation of the query during the matching phase. Thus, a good indexing must make it possible to find only the relevant documents.

The indexing process can be performed in three different modes: (i) *Manual* (made by a human): a domain expert who examines and analyzes each document in the collection to select the best representative keywords. This type of indexing guarantees a better representation of the documents and a better quality of the results. However, manual indexing requires a substantial amount of time. (ii) *Automatic* (using a computer process): this process of indexing is fully

automated. This process includes the following steps: automatic extraction of terms, elimination of stop words, normalization of extracted terms, weighting of terms and finally the creation of the index. (iii) *Semi-automatic* (combination of both modes): It is based on the results obtained by automatic indexing. Nevertheless, an intervention of an expert or specialist can be carried out in order to enrich and choose the significant terms, in order to validate the final representation of the documents and the query.

As part of our work, we are interested in the automatic indexing process. Below, we present the different steps of automatic indexing:

Automatic extraction of terms:
This step divides the document into a set of terms (often called Tokens). A term is considered as a sequence of characters separated by blanks or special characters.

Elimination of stop words:
A stop word is a non-useful term, such as articles, personal pronouns, prepositions, etc. Stop lists are used to filter and eliminate those non-useful terms. The elimination of these stop words is a fundamental step in automatic indexing.

Normalization of extracted terms:
Once the words are extracted, they will be normalized using stemming. Stemming (Lemmatization) is a morphological process to find a more general form of words.

Term weighting:
Term weighting determines the importance of terms in describing the content of a given document. The term weighting techniques are based on the notions of frequency of terms in a document (tf, Term Frequency: a measure that indicates the importance of the term in the document - local weighting) and frequency of these terms in the corpus (idf, Inverse Document Frequency: a measure that indicates the importance of the term in the whole collection -

global weighting). The combination of these two measures ($tf.idf$) gives an approximation of the importance of term in the document and the corpus. One popular $tf.idf$ term-weighting function is described within Okapi BM25 model and given by Formulas 2.1 and 2.2:

$$tf = \frac{occ_i}{k_1 \left((1-b) + b\frac{dl}{avdl} \right) + occ_i} \tag{2.1}$$

Where:

occ_{ij}, is the frequency of the term t_i in a document d;

k_1 and b, are constants;

b, is a constant;

dl, is the document length;

$avdl$, is the average of document length;

$$idf = \log \frac{N - n_i + 0.5}{n_i + 0.5} \tag{2.2}$$

Where:

N, is the number of documents in the whole collection;

n_i, is the number of documents in the collection containing t_i.

The final Okapi BM25 term-weighting function is therefore given by Formula 2.3:

$$w_i^{BM25} = \frac{occ_i}{k_1 \left((1-b) + b\frac{dl}{avdl} \right) + occ_i} \log \frac{N - n_i + 0.5}{n_i + 0.5} \tag{2.3}$$

Regarding the internal parameters, a considerable number of experiments have been done and suggested that values such as $1.2 < k_1 < 2$ and $0.5 < b < 0.8$ are reasonably good in many cases. The study of Robertson and Zaragoza [62] indicated that published versions of Okapi BM25 are based on specific values assigned to k_1 and b: $k_1 = 2, b = 0.5$.

Index creation:
In the final phase of automatic indexing, terms are stored in data structures. These structures allow efficient access to the representation of documents. One of the most used structures is the inverted index. An inverted index indicates which documents that particular term occurs in.

b. Matching

The matching process is triggered whenever a user submits a new query to the information retrieval system. This process describes the following scenario: (i) the user expresses his need for information in the form of a query, and then (ii) the system processes the query and evaluates the relevance of the documents to that query using a document-scoring function. Finally, (iii) the system returns a list of documents ordered from the most relevant to the least relevant.

Matching function:
The matching function relates the terms of a document with those of a query by establishing an equality relation. In other words, the matching function consists in determining the relevance of a document to a query and ranking the retrieved ones in order of relevance. The degree of relevance is calculated using a document-scoring function denoted $RSV(Q, d)$, Retrieval Status Value, where Q is a query and d is a document. One popular document-scoring function is described within Okapi BM25 model and given by Formula 2.4:

$$RSV(Q, d) = \sum_{t_i \in Q \cap d} w_i^{BM25} \tag{2.4}$$

Where:

w_i^{BM25}, is the weight of term t_i in document d.

The matching process is intimately related to the process of indexation and weighting of terms. Indeed, the representation of the documents and the queries and their matching define an information retrieval model.

2.2.2 IR models

An information retrieval model plays a very important role in an IRS. It consists of: (i) providing a formalism to represent documents and queries based on the weighted terms derived from indexing. (ii) Determining a method of comparison to calculate the degree of correspondence between the representation of the documents and the representation of the queries. Several information retrieval models have been proposed and developed since the birth of the first IRS. These models have in common the indexing vocabulary, and differ mainly by the matching model. We present here the main information retrieval models: the Boolean model, the vector space model and the probabilistic model.

a. Boolean model

The Boolean model is the oldest model in the field of information retrieval. This model is mainly based on set theory, where the query terms are either present or absent, and the document is either relevant or irrelevant.

Documents and queries are characterized by sets of terms, or rather by Boolean vectors. Each document is represented by a logical conjunction of unweighted terms.

Let d be a document:

$$d = t_1 \wedge t_2 \wedge t_3 \wedge ... \wedge t_n \tag{2.5}$$

Where:

t_i, is a term ($i = 1, 2, ..., n$)

The query is represented as a logical expression. In this expression, the terms are connected by Boolean operators $\vee$ (OR), $\wedge$ (AND) and $\neg$ (NOT). These operators are used to perform union, intersection and difference operations.

Let Q be a query:

$$Q = t_1 \wedge \neg(t_2 \vee t_3) \tag{2.6}$$

The documents that satisfy the logical query's expressions are considered as relevant. The score of each document is calculated using the Boolean RSV function, which measures the degree of correspondence between the query Q and the document d. The output of this function is a binary value and described as follows: $RSV(Q, d) = \{1, 0\}$.

Although the standard Boolean model is simple and fast to implement and provides a very clear and expressive query language, it has a number of weaknesses, such as: (ii) the relevance of the documents is a Boolean variable, which does not support ranking the returned results in order of relevance. (ii) Formulating the query with several logical operators (AND, OR, NOT) is too complicated. (iii) The notion of 'uncertainty' of information cannot be modeled with logical operators.

In order to remedy these drawbacks, several extensions have been introduced, among them there are: (i) the *extended Boolean model*: it is an improvement of the standard Boolean model. It takes into account the weights of terms in the documents so that the returned documents can be ranked according to their correspondence to the query. (ii) The *fuzzy Boolean model*: it is based on the theory of fuzzy sets. The objective is to represent the uncertainty of the information. It allows describing a term by a degree of membership to a fuzzy set and characterizing a document as a fuzzy set of terms.

b. Vector Space model

Several variants of the vector space model have been implemented in many IRS. This model is based on an algebraic approach, which represents the content of the documents and the query in the same vector space. The dimensions of the vector space are the set of vocabulary terms resulting from the indexing process.

In this model, each document is represented by a vector in a multidimensional space, where each dimension corresponds to a term weight, as follows:

Let d be a document:

$$d =< w_1, w_2, w_3, ..., w_n > \tag{2.7}$$

Where:

w_i, is the weight of the term t_i in document d.

In the same way, the query is represented as a vector of the weights of the terms:

Let Q be a query:

$$Q =< v_1, v_2, v_3, ..., v_n > \tag{2.8}$$

Where:

v_i, is the weight of the term t_i in document Q;

$i = 1, 2, ..., n.n$: is the size of the vocabulary.

The relevance of document d with respect to query Q is evaluated by the degree of similarity, denoted $RSV(Q, d)$, between the vector of the document and that of the query. The most commonly used similarity measures are:

The Scalar Product:

$$RSV(Q, d) = \sum_{i=1}^{n} v_i * w_i \tag{2.9}$$

The Cosine measure:

$$RSV(Q, d) = \frac{\sum_{i=1}^{n} v_i * w_i}{\sqrt{\sum_{i=1}^{n} (v_i)^2 * \sum_{i=1}^{n} (w_i)^2}^{\frac{1}{2}}} \tag{2.10}$$

The Jaccard measure:

$$RSV(Q, d) = \frac{\sum_{i=1}^{n} v_i * w_i}{\sum_{i=1}^{n} {v_i}^2 + \sum_{i=1}^{n} {w_i}^2 - \sum_{i=1}^{n} v_i * w_i} \tag{2.11}$$

The vector space model has several advantages over the standard Boolean model, such as: (i) the non-binary weighting of the terms provides superior quality of the results and allows returning documents that partially respond to the query. (ii) The formulation of the query is in natural language and closer to the user needs. (iii) The matching function is used to rank the results according to their level of relevance to the query.

The vector space model also has several limitations, such as: (i) the independence between the terms, and (ii) this model does not take into account the semantic aspect of terms.

c. Probabilistic model

The probabilistic model is based on probability theory. It is modeled by two well-known models: (i) the BM25 model based on the Okapi search engine and (ii) the 2-Poisson model. It allows the modeling the notion of relevance, by estimating the probability of relevance of a document with respect to a query through the "Probability Ranking Principle". The basic principle of probabilistic models consists in selecting and sorting the documents in descending order of the probability of relevance to the query. These documents must have at the same time a high probability of being relevant, and a low probability of being irrelevant.

For a query Q, the set of documents is divided into two groups: Relevant documents (R) and irrelevant documents ($\bar{R}$). Assuming the independence of terms, two conditional probabilities are associated with each document d:

$P(R|d)$: The probability that a document d is relevant to a query Q. $P(\bar{R}|d)$: The probability that a document d is irrelevant to a query Q.

The relevance of the document d with respect to the query Q is evaluated by the matching function, denoted $RSV(Q, d)$, as follows:

$$RSV(Q, d) = \frac{P(R|d)}{P(\bar{R}|d)} \tag{2.12}$$

The probabilistic model has several advantages: (i) this model is more efficient than the standard Boolean model. (ii) It effectively addresses the uncertainty of information. The probabilistic model also has a number of disadvantages, such as: (i) it is less efficient than the vector space model. (ii) There is no method of estimating the relevance of terms.

2.2.3 IR Evaluation

Evaluation is a necessary process in an information retrieval system. This process consists of evaluating the quality of the results returned by an SRI, comparing its performance with other SRIs, and also validating IR models and techniques.

To evaluate an information retrieval system, it will be sufficient to submit the queries and then compare the returned results against the expected results. The difference between the results of the system and the expected results is then calculated to measure the performance of the system.

The evaluation of the performance of an IRS is performed using two resources: (i) the test collection and (ii) the evaluation measures. The test collection consists of three main elements: (i) a collection of documents, (ii) a set of queries, and (iii) a set of judgments (documents judged relevant to each query). Regarding the evaluation measures, precision and recall measurements are the most used.

a. Test collection

A test collection is used to evaluate the performance of an information retrieval system. It is composed of the following elements: (i) *Corpus of documents*: it is a set of documents to be indexed, from which the system will be evaluated. (ii) A *set of queries (topics)*: this is a set of information needs used for the test. (iii) *Judgments of relevance*: this is the list of documents relevant to each query. The realization of these judgments is a long and costly task involving humans.

The test collections are generally the results of conferences and forums, such as TREC (Text REtrieval Conference) and CLEF (Cross-Language Evaluation Forum).

b. Efficiency measures

In information retrieval, it is so important that the amount of data processed by an IRS is high, whereas the response time is low. Therefore, the efficiency of an IRS consists in providing fast and ordered access to large amounts of information. It ensures that information retrieval systems adapt to the vast amounts of information to be retrieved. Over the past years, researches on efficiency have mainly focused on efficient indexing, storage and retrieval of data.

The efficiency can be measured in both the computational cost and the memory space consummation. The CPU time and the complexity of IR algorithm (in terms of the number of operations) can be used to compute the computational cost, whereas, the indexes sizes can be used to evaluate the memory space consummation.

c. Effectiveness measures

As previously described, the evaluation of information retrieval systems is an important step in the development of an information retrieval model. It characterizes the model and provides elements of comparison between the models.

The main objective of an information retrieval system is to select all relevant documents and reject all irrelevant documents. This objective is evaluated using different statistical metrics, namely: recall, precision, and mean average precision.

Recall:

The recall measures the ability of an IRS to select relevant documents. This

measure calculates the ratio of relevant documents selected (RelS) to total number of relevant documents available (Rel):

$$Recall = \frac{RelS}{Rel} \tag{2.13}$$

Precision:
Precision measures the ability of an IRS to select only relevant documents. This measure calculates the ratio of relevant documents selected (RelS) to total number of selected documents (S):

$$Precision = \frac{RelS}{Rel} \tag{2.14}$$

An ideal IRS is a system that returns all relevant documents (recall = 1), and all the returned documents are relevant (precision = 1).

Mean average precision (MAP):
This measure calculates the average of the average precision values for each relevant document. It computes the precision of the results with respect to a set of queries and focuses primarily on the top ranked relevant documents:

$$MAP = \frac{1}{NQ} \sum_{i=1}^{NQ} \frac{1}{m_i} \sum_{j=1}^{m_i} P(R_{ij}) \tag{2.15}$$

Where:

NQ, is the number of queries;

m_i, is the number of relevant documents for the ith query;

R_{ij}, is the set of ranked retrieval results from the top result until the document d_j is achieved.

2.3 IR and query expansion

The exact vocabulary of a document that satisfies the user information need is often not reflected by the user initial queries. In addition, a document might

contain the exact information that a user is searching but it cannot be retrieved, as it does not contain any of the keywords used in the user's query.

The users' initial queries often do not express the information needs in a satisfactory manner. This poor quality of users' queries is due to: (i) users express their needs with short queries. (ii) Users employ ambiguous and imprecise keywords to characterize their needs. (iii) Users do not have precise ideas about the collection of documents on which they run their queries (iv) Non-expert users do not use the parameters provided by the search model to better express their queries.

A query expansion approach is therefore often necessary in order to benefit as much as possible from the capabilities provided by an information retrieval system. Expanding a query consists in enriching the user's initial query with additional information, so that the system generates and returns more appropriate results to satisfy the user's needs.

Usually, query expansion techniques depend on three factors: (i) the collection of documents used to find the expansion term candidates. (ii) The method adopted to generate and select the best expansion terms. (iii) The manner how the expansion terms are integrated into the initial query.

2.3.1 Query expansion Techniques

Information retrieval systems offer different query expansion techniques to refine and improve the user initial queries. Most of these techniques are based on statistical analyzes, linguistic resources, or using terms from the relevant documents.

a. Query expansion based on vocabulary

This category of query expansion techniques relies on the vocabulary used in the search process. Many systems have used statistical analyzes such as the calculation of co-occurrence between terms. Other systems have used linguistic resources such as semantic relations between terms.

Linguistic techniques:

Many query expansion techniques have used linguistic relationships between terms to improve search results. These techniques are mainly based on: (i) the morphological family of the word: indeed, each word has a morphological family. It is therefore relevant to extend a research performed on a term with all the words of its morphological family. (ii) Semantic relations between terms: the main semantic relationships used for query expansion are the synonymy, hyperonymy and hyponymy relations.

These techniques often use thesaurus to construct term families and retrieve semantically related words. A thesaurus is a data structure that gathers several linguistic information and describes concepts and the semantic relationships that exist between them. The construction of this type of thesaurus is usually done manually.

The thesaurus is therefore used to select and extract the terms that are strongly related to the original query terms. A weight is then assigned to each extracted term, which will be added to the original query to form a new query. One of the most widely used thesaurus for query expansion is: *WordNet*

WordNet is a lexical thesaurus designed manually by experts in order to list, classify and relate in various ways the semantic content of the English language. WordNet distinguishes four grammatical groups: names, adjectives, verbs and adverbs. These are organized as sets of synonyms called Synset. The sets of synonyms are linked together by different types of relationships, such as: synonymy, antonymy, hyperonymy, etc. Most of the time, WordNet was used during the query expansion process to add synonyms to the original query.

Expanding the original query using WordNet has several advantages, such as: (i) succeed to improve the performance of bad queries such as short queries. (ii) succeed to increase the recall. The use of WordNet also has several limitations, such as: (i) for long queries, the performance tends to degrade. (ii) WordNet is missing most of the relationships between terms that are useful for information retrieval. (iii) It degrades the precision.

Statistic techniques:

Statistical techniques are based on statistical analysis of the corpus. They aim to find associations of terms that tend to appear together in documents. These associations are usually created automatically and based on the co-occurrence of the terms in documents, i.e. the fact that two terms often appear together in the same document. The analysis of co-occurrence between terms, on a collection of documents, allows finding the pairs of words being strongly related.

Mutual information is often used to estimate the co-occurrence between terms in a corpus of documents. It compares the probability of the co-occurrence of terms x and y with the independent probabilities of occurrence of x and y. Mutual information $I(x, y)$ is given by Formula 2.16:

$$I(x, y) = \log_2 \left[\frac{P(x, y)}{P(x)P(y)} \right] \tag{2.16}$$

Where:

$P(x, y)$, is the probability of occurrence of both terms x and y within a certain context;

$P(x)$ and $P(y)$, are the probability of occurrence of terms x and y, respectively. Term frequency can be used to estimate such probabilities.

From formula 2.16, it can be realized that the mutual information is symmetric. The major issue with asymmetric mutual information is that the rare terms are favored against common terms. To overcome this issue, the conditional probability can be used to compute the strength of the co-occurrence of term y to term x, as shown in formula 2.17:

$$P(x|y) = \frac{P(x, y)}{P(x)} \tag{2.17}$$

In formula 2.17, the probability $P(x, y)$ can be expressed by the number of phrases or sentences that include both terms x and y; whereas, the probability $P(x)$ can be characterized by the number of phrases or sentences in which term x occurs.

Although statistical techniques are simple to set up within an information retrieval system, they have a number of weaknesses, such as: (i) these techniques are not as successful as linguistic techniques. (ii) They still remain very costly on today's corpus. (iii) In these techniques, terms that have a high degree of co-occurrence are very common terms in the collection of documents, and therefore they can affect the retrieval performance of query expansion.

b. Query expansion based on Judgments

Another category of query expansion is based on the use of terms selected from documents that have been judged to be relevant. There are two methods that use terms from relevant documents: (i) *Relevance Feedback* (manual judgments): In this technique, the user manually examines the returned documents and selects the relevant documents. (ii) *Pseudo-Relevance Feedback*: In this technique, the system automatically considers the top-ranked documents, retrieved in response to the original user query, as relevant documents.

Manual Judgments

Relevance feedback is an effective method to improve the retrieval effectiveness of IRS. This technique allows users to review the results of their original query by reporting relevant and/or irrelevant documents. It works as follows: (i) first, a preliminary search is required as the source of the expansion terms is provided by the relevant documents. This step allows selecting a set of relevant documents from among the list of documents judged by the user. (ii) then, from these relevant documents, the best expansion terms are selected. This step consists in extracting the relevant terms that will be used to enrich the initial query. (iii) finally, add the expansion terms to the original query and re-weight the query terms. This step consists in constructing a new query by combining the initial query with the information extracted in the preceding step.One of the best term scoring functions are:

Robertson/Sparck Jones weight (RSJ) [63]:

$$w_i^{RSJ} = \log \frac{(r_i + 0.5)(N - R - n_i + r_i + 0.5)}{(n_i - r_i + 0.5)(R - r_i + 0.5)} \tag{2.18}$$

Rocchio's weight [64]:

$$w_i^{Rocchio} = \sum_{d \in R} w_i \tag{2.19}$$

Where:

N, is the number of documents in the whole collection;

n_i, is the number of documents in the collection containing expansion term candidate t_i;

R, is the number of documents judged relevant;

r_i, is the number of judged relevant documents containing expansion term candidate t_i.

The main interest of using Relevance Feedback is given by: (i) its simplicity and performance. (ii) Only the terms contained in the documents returned by the system and judged relevant by the user are considered. However, this method is still unpopular amongst Internet users.

Automatic Judgments

Pseudo-Relevance Feedback (PRF), called also Blind Feedback or Retrieval Feedback, is another query expansion technique that selects expansion terms from documents that have been judged relevant. This technique consists in considering the top-ranked documents retrieved in response to the original user query as pseudo-relevant documents.

In its simplest version, the pseudo-relevance feedback starts by: (i) performing an initial search on the original query. The top retrieved documents are considered as pseudo-relevant documents. (ii) Then, extracting the expansion keywords from the pseudo-relevant documents and ranking them using a term-scoring function such as: Robertson/Sparck Jones term-ranking function (RSJ) and Rocchio weight. (iii) Finally, expanding the original query by

adding the top ranked keywords and retrieving the relevant documents to the expanded query using a document-scoring function.

Although Pseudo-Relevance Feedback (PRF) techniques improve the performance of information retrieval systems, they can also degrade the quality of results if the pseudo-relevant documents contain little or no relevant information.

2.3.2 Related work, QE in Web IR

Query expansion has made a big leap forward by developing effective techniques that bypass the difficulty of providing more precise description of user request. Moreover, it has yielded considerable improvements in retrieval performance over the conventional information retrieval systems. However, the latest proposed query expansion techniques for Web IR systems failed to enhance the retrieval effectiveness and returned irrelevant information [12]. This is predominantly due to the fact that all of these techniques are based on traditional modelling of query expansion, which is mainly seeking for the best expansion keywords and not really the appropriate expanded query.

a. Semantic approaches

Most of these QE techniques involved semantic resources and approaches to find the suitable expansion keywords. For instance, Qiang et al. [52] attempted to expand the initial query using knowledge expansion words derived from an external semantic resource for Web information retrieval called Freebase. Li et al. [49] explored also the Freebase resource for query expansion. The authors used the Freebase to discover the global and local expansion entities that are related to the initial query and the top ranked pseudo-relevant documents. Leung et al. [45] studied several semantic approaches to concept-based query expansion and compared them with different ontology-based expansion methods in Web information retrieval. The main idea was to measure the distance between candidate expansion concepts using a collaborative semantic

proximity measure called PMING distance. Brandao [11] suggested an entity-oriented query expansion process that exploits semantic sources of evidence to devise discriminative term features. The author involved machine learning techniques to combine these features in order to select and rank candidate expansion terms. Biancalana et al. [8] introduced a semantic technique to select the best expansion terms. Contrary to traditional query expansion techniques which based on the computation of two-dimensional co-occurrence matrices, the proposed technique used three-dimensional matrices, where the added dimension is represented by semantic classes related to the folksonomy extracted from social bookmarking services. Still in the same direction, El Ghali et al. [18] presented a context-based query expansion method for Web short queries. The authors selected the best expansion keywords using Latent Semantic Analyses (LSA) method which based on the result of the three following query suggestion methods: the Cosine Similarity (CS), the Language Models (LM), and their fusion. A context-aware query expansion method is another work that based on LSA method [17]. In this work, the user's initial query is enriched by additional expansion contexts to build the global query context. Another related work is the one introduced in [70]. In this work, the term co-occurrence information and semantic information of term were combined to rank the expansion keywords obtained from the top feedback documents. Anand and Kotov [3] used term graphs constructed from Web collections as well as external resources, such as encyclopedias (DBpedia) and knowledge bases (ConceptNet), as sources of semantically related terms for query expansion. Jain et al. [28] investigated the role of graph structure for query expansion. The authors determined the importance of each node in the graph using the semantic ontology WordNet. The relevant nodes representing word senses are identified from the graph and were chosen as additional expansion terms to be added to the original query in order to improve the retrieval of web pages.

b. Evolutionary algorithms and swarm intelligence

Evolutionary algorithms and swarm intelligence algorithms were also employed to select the best expansion keywords for Web information retrieval. Bhatnagar and Pareek [7] developed a genetic algorithm-based query expansion to find the appropriate expansion terms and improve the retrieval performance of Web IR systems. In this method, the top ranked candidate keywords constitute a term pool. From this term pool, genetic algorithm is used to select a combination of terms that will be added to the initial query. Khennak and Drias [42] designed a bat algorithm based query expansion for Web medical information retrieval. The authors used bat algorithm to retrieve simultaneously the appropriate expansion keywords and the best relevant documents to the expanded query. Particle swarm optimization algorithm was also introduced to improve QE through choosing the appropriate combination of weights of terms in query vector and used fitness function as a measure of the proximity between the reweighted query vector and top ranked documents retrieved from original query vector [9]. Gao and Xu [21] exploited Web search logs for query expansion using random walks. In this work, QE was based on path-constrained random walks, where the search logs are represented as a labeled, directed graph, and the best expansion terms were selected by a learned combination of constrained random walks on the graph.

c. Social media

Many approaches were also proposed to select the best expansion words in Web social media. Miyanishi et al. [54] involved QE to improve the retrieval effectiveness of Twitter search. The proposed approach tried to extract the suitable expansion words using two-stage relevance feedback that models search interests by manual tweet selection and integration of lexical and temporal evidence into its relevance model. Geng et al. [22] proposed a query expansion method for social book search to expand and enrich the social information of queries. In this method, the authors selected the high-frequent words in the pseudo-relevant document to be used as expansion terms. They also employed

linear smoothing to combine the original query terms and the best selected expansion keywords. A personalized query expansion for Web search using social keywords was described by Sarwar et al. [67]. In this work, the authors suggested to model the social context of the person as the status messages generated by the socially associated people and to use his social context to improve the web search query expansion process. Zhou et al. [92] tried also to enhance the Web information retrieval via personalized query expansion using social media. The authors suggested a QE framework based on individual user profiles mined from the annotations and resources the user has marked. In this framework, the most appropriate expansion terms for a query are likely to be associated with and influenced by terms extracted from the documents ranked highly for the initial query. Jung [33] sought to enhance query expansion results by analyzing a multilingual social tagging which was written in several different languages. In this study, a co-occurrence measurement was used to find meaningful relationships between tags from a large scale social tagging. The discovered association patterns among multilingual tags were then explored to query expansion for better information searching.

d. Medical and health

Query expansion was involved in Medical field. For instance, Lu et al. [51] investigated the retrieval performance of PubMed's Automatic Term Mapping process. This process uses an automatic query expansion method to enrich the original search query with additional terms extracted from PubMed's translation tables (MeSH table, journals translation table and author index). A semi-automatic semantic based query expansion approach was proposed by Curé et al. [15] to refine health outcomes of interest. In this approach, query expansion was adopted to generate the appropriate health outcomes of interest (HOI) from a large number of medical and health terminologies. It used a lattice of concepts created over hundreds of ontologies and terminologies to find the best matching concepts through leveraging the hierarchical structure of ontologies and then expand the search to include the appropriate HOIs. Jalali and

Borujerdi [30] developed a concept based pseudo-relevance feedback to improve the performance of biomedical retrieval systems. The proposed method used a hybrid semantic retrieval algorithm to assess the degree of relevance between search queries and documents. It also used a retrieval feedback mechanism to expand the original search queries with additional biomedical concepts extracted from top ranked results of hybrid semantic retrieval algorithm. Jain et al. [29] suggested an Electronic Medical Record (EMR) retrieval system that leverages query expansion to retrieve more relevant medical records to the patient's current symptom. The suggested framework integrated information retrieval techniques with medical domain knowledge available in domain ontologies and extracted from healthcare expert. It also employed several sources of knowledge including semantic relationships and heuristics, such as synonyms and term co-occurrence, in order to expand and enhance initial search queries.

e. Other approaches

A simple term frequency transformation model [87] is another stream of work which based on traditional modelling of QE. This work proposed a simple and heuristic model, in which three term frequency transformation techniques were integrated to capture the saliency of a candidate term associated with the original query terms in the feedback documents. Singh and Sharan [71] suggested a fuzzy logic-based query expansion model for Web information retrieval. The suggested model employed multiple terms selection methods together to extract the appropriate expansion terms. It combined different weights of each term through using fuzzy rules to infer the weights of the expansion terms. Leturia et al. [44] sought to enhance the Basque Web retrieval through QE technique. The authors used morphological query expansion and language filtering words in combination with the APIs of search engines to choose the most appropriate expansion keywords and build the suitable Basque Web search queries.

Chapter 3

Nature-Inspired Optimization

Swarm intelligence (SI) and bio-inspired computation have received great interest and attention in the literature. In the communities of optimization, computational intelligence, and computer science, bio-inspired algorithms, especially those SI-based algorithms, have become very popular. In fact, these nature-inspired Metaheuristic algorithms are now among the most widely used algorithms for optimization and computational intelligence. SI-based algorithms such as ant and bee algorithms, particle swarm optimization, cuckoo search, and firefly algorithms can possess many advantages over conventional hard computing algorithms.

In this chapter, we first introduce Nature-Inspired algorithms (Section 3.1) and then present three powerful swarm intelligence algorithms: Bat Algorithm (Section 3.2.1), Firefly Algorithm (Section 3.2.2) and Accelerated Particle Swarm Optimization (Section 3.2.3)

3.1 Nature-Inspired Metaheuristics

Almost all standard and traditional algorithms, such as the simplex method in linear programming, are deterministic. Certain of deterministic optimization algorithms, such as the popular Newton-Raphson algorithm, depend on the gradient information. This class of algorithms are called *gradient-based algorithms*.

Generally, we have two kinds of stochastic algorithms: Heuristic and Metaheuristic. In heuristic algorithms, we do not look for the optimal solutions, but

rather, we seek for the good solutions that can be reached rapidly. In other words, in heuristic algorithms, there is no guarantee to find the optimal solutions but the acceptable solutions can be reached in a reasonably practical time.

Heuristic algorithms can generate feasible solutions to a complex problem in an acceptable timescale. These algorithms do not ensure finding best solutions. The goal is to have efficient and practical algorithm that will perform most the time and that is able to reach good solutions in a reasonable amount of time.

Metaheuristic algorithms, which are further development of heuristic algorithms, can do better than heuristics. They are often employed for global optimization. These algorithms are based on both local search and randomization. This latter characteristic allows Metaheuristic to move from local scale to global search.

Metaheuristic algorithms should have two essentials components: exploitation and exploration. The exploitation, or intensification, concentrates the search in a local area through exploiting the information that a current good solution is found in. On the other hand, exploration, or diversification, focuses on producing various solutions in order to explore the search space on a global scale. The exploitation generally leads to very fast convergence towards the optimal solution. The exploration in contrast enhances the diversity of solutions and avoids the solutions to get stuck at local optima. The global optimality can be reached when these two components are appropriately combined.

Metaheuristic algorithms can be classified as population-based or trajectory-based. For instance, Particle Swarm Optimization (PSO), Firefly Algorithm (FA), and Cuckoo Search (CS) are population-based as they involve a set of agents or particles. Genetic Algorithms are also population-based, which use a set of strings. In contrast, Simulated Annealing (SA) is a trajectory-based. It uses a single solution which moves across the search space. In this class of algorithms, the better solution is always accepted, whereas, the no-good solution is not always accepted.

3.1.1 Exploration and exploitation

From the point of view of search space exploration, swarm intelligence algorithms have to guarantee two essentials features: *exploitation* and *exploration*.

a. Exploitation

In order to produce new solutions that are better than the current ones, exploitation feature may involve any knowledge acquired from the problem of interest. By definition, exploitation, which is also referred to as *intensification*, is for local search. It is based on local information. Although, this feature can stick in a local mode due to the final solution which can be affected by the starting point, exploitation generally guides to very fast convergence rates.

b. Exploration

Exploration, which is also referred to as *diversification*, is for global search. It allows the generation of divers solutions that can be far from the existing ones. As a result, the search space can be explored more efficiently. Although, this feature can lead to slow convergence rates, as well as, consume heavy computational cost to handle several inappropriate new solutions that are far from the global optimality, exploration usually does not stick in a local optimum.

Finding the global optimality can be hard when the exploitation is too high and the exploration is too low. However, the system can converge more rapidly in this case. In contrast, when the exploitation is too low and the exploration is too high, the system can converge very slow. Accordingly, in order to achieve optimal performance, swarm intelligence algorithms have to find the optimal balance between exploitation and exploration.

Achieving the optimal balance is still a challenging issue. Many studies addressed this issue and attempted to solve it. Nevertheless, they failed to find the typical balance. This is due to the fact that the balance between exploitation and exploration varies from problem to problem. Moreover, it depends on several

criteria including algorithm mechanism, parameters setting, etc. Thus, finding the appropriate balance can be itself considered as an optimization problem.

3.2 Swarm intelligence algorithms

In the last two decades, a new kind of nature-Inspired algorithms has emerged, referred to as Swarm Intelligence Algorithms. These algorithms are stochastic search methods that were inspired by the social behaviors of insect and animal herds in nature [82, 39]. The power of these algorithms lies in their capability of handling single and multi-objective, unconstrained and constrained optimization problems. For instance, Particle Swarm Optimization (PSO) was designed based on the social foraging behavior of some animals such as bird flocking and fish schooling [38], while the Firefly Algorithm (FA) was inspired by the flashing behavior of fireflies and the phenomenon of bioluminescent communication [79]. New algorithms are also emerging recently, including Cuckoo Search (CS) and Harmony Search (HS) [84]. In the following, we present three promising swarm intelligence algorithms which are later involved to improve the retrieval effectiveness and computational complexity of query expansion.

3.2.1 Bat Algorithm

The Bat Algorithm (BA), developed by Yang in 2010 [83], is one of the recent swarm intelligence algorithms which has been proved very efficient. BA is based on natural bats echolocation simulation and designed by mimicking bats foraging behavior. Bats are the only mammals that possess wings. There are around 1000 different species of bats, most of them are insectivore. There are two types of bats: the mega-bats and the micro-bats, which differ by mainly their size but also by the echolocation phenomenon, which is more present in micro-bats.

a. Standard Bat Algorithm

BA is based on the echolocation of microbats, which use a type of intelligent sonar (echolocation) to communicate, recognize different types of insects, sense distance to their prey and move without hitting to any obstacle even in complete darkness. These microbats emit a very loud sound pulse and detect the loudness variations of the echoes to build up a three dimensional scenario of the surrounding. They can detect the distance and orientation of the target, the type of prey, and even the moving speed of the prey, such as small insects. Based on the description and characteristics of bat echolocation, the Bat Algorithm was developed with the following three rules:

- All bats use echolocation to sense distance, and they also 'perceive' the difference between food/prey and background barriers in some magical way.

- Bats fly randomly with velocity v_i at position x_i with a frequency $f_i \in [f_{min}, f_{max}]$, varying wavelength λ and loudness A_0 to search for prey. They can automatically adjust the wavelength (or frequency) of their emitted pulses and adjust the rate of pulse emission $r \in [0, 1]$, according to the proximity of their target.

- Although the loudness can vary in many ways, we assume that the loudness varies from a large (positive) A_0 to a minimum constant value A_{min}.

b. Bat motion

Apart from the control parameters, such as the population size and maximum iteration number which are common control parameters for all nature inspired algorithms, the BA has few important parameters to control. The frequency parameter is one of these important parameters. It is similar to the key feature used in the PSO and HS. This parameter serves for automatically zooming into a region, where the promising solutions have been found and balances automatically from exploration to exploitation. This gives superiority to the BA over the other evolutionary algorithms in the literature.

In simulation, at iteration t, each bat in the swarm moves randomly with a velocity v_i^t and a position x_i^t. The position of each bat (solution) is evaluated by calculating its fitness function value $f(x_i)$ and the overall best solution x_* is assigned according to this value. The position x_i^{t+1} of each solution in the swarm is adjusted by moving virtual bat x_i^t according to Equation 3.1:

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{3.1}$$

Where:

v_i^{t+1}, is the velocity of movement which is calculated by Formula 3.2:

$$v_i^{t+1} = v_i^t + (x_i^t - x_*)f_i \tag{3.2}$$

Where:

x_*, is the current best solution;

f_i, is the frequency. It is computed using Equation 3.3:

$$f_i = f_{min} + (f_{max} - f_{min})\beta \tag{3.3}$$

Where:

$\beta \in [0, 1]$, is a random vector drawn from a uniform distribution;

f_{min} and f_{max}, are the specified lower and upper bounds for the frequency parameter f_i.

c. Loudness and pulse emission rate

The loudness A_i and the pulse emission rate r_i are very important parameters in a Bat Algorithm. The loudness decreases once a bat has found its prey, while the pulse emission rate increases. The value of loudness can be updated during the search as shown in Equation 3.4:

$$A_i^{t+1} = \alpha A_i^t \tag{3.4}$$

Where:

α, is a constant and plays a similar role as the cooling factor of a cooling schedule in the simulated annealing and this way, it prevents from premature convergence. In addition, the pulse emission rate parameter can be updated using Equation 3.5:

$$r_i^{t+1} = r_i^0 \left[1 - e^{-\gamma t}\right] \tag{3.5}$$

Where:

$\gamma > 0$, is a constant;

r_0^i, is initial pulse rate in the range $(0, 1]$.

d. Local search

Each bat (solution) in the population is evaluated by calculating its fitness function value and the overall best solution is selected as a current best solution x_*. Once the best solution is selected, each bat in the population generates a new solution through a random walk as shown in Formula 3.6:

$$x_{new} = x_{old} + \epsilon A^{(t)} \tag{3.6}$$

Where:

$\epsilon \in [-1, 1]$, is a random number;

$A^{(t)}$, is the average loudness of all the bats at this time step.

The random walk strategy allows the algorithm to balance between exploration and exploitation.

Through these mathematical formulas, we can observe that BA degenerates into PSO when we replace the frequency f_i by a random parameter and set $A_i = 0$ and $r_i = 0$. In addition, when we do not use the velocities and fix A_i and r_i, it degenerates into Harmony Search (HS). In other words, PSO and HS can be considered as special cases of Bat Algorithm. Therefore, BA has the advantages of these two algorithms.

e. The original Bat Algorithm

The main steps of the Bat Algorithm are given in Algorithm 1 and its description can be summarized as follows:

Algorithm 1 pseudo code of the Bat Algorithm (BA)

```
Fitness function f(x)
Initialize the bat population x_i and v_i (i = 1, 2, ..., N)
Initialize frequencies f_i, pulse rates r_i and the loudness A_i
while (t < Max number of iterations) do
    Generate new solutions by adjusting frequency (Equation 3.3);
    Update velocities and locations/solutions (Equations 3.1 and 3.2);
    if rand[1] > r_i then
        Select a solution among the best solutions;
        Generate a local solution around the selected best solution (Equation 3.6);
    end if;
    Generate a new solution by flying randomly;
    if rand < A_i and f(x_i) < f(x_*) then
        Accept the new solutions;
        Increase r_i and reduce A_i (Equations 3.4 and 3.5);
    end if;
    Rank the bats and find the current best x_*;
end while
```

The algorithm starts by setting the initial values of its parameters. It firstly generates the initial population of N solutions randomly $(i = 1, 2, ..., N)$, where each solution (bat) is initialized with a velocity v_i^0, a location x_i^0 and assigned an initial frequency f_i. The values of pulse rate r_i and loudness A_i are initialized and the initial population is evaluated by calculating the value of the objective function for each solution $f(x_i^0)$. The best solution x_* is then initialized by the best solutions of all the bats. The BA iteratively moves from the initial set of solutions to the best solution by adjusting the frequency f_i, the position x_i and the velocity v_i of each artificial bat as shown in Equations 3.1, 3.2 and 3.3. The new solutions are evaluated and the best solution x_* is updated. The local search method is applied by using a random walk method as given in Equation 3.6 in order to refine the best-found solution. As bats get closer to its new

[1] $rand$ is a function that generates a random real number in the [0,1] interval

better solutions, the pulse emission rate and loudness are updated gradually using Equations 3.4 and 3.5. Solutions are continuously updated based on the continuous flying iterations. This is repeated till the termination criteria are satisfied. Finally, when all criteria successfully met, the best so far solution is reported.

f. Applications

Nowadays, BA has attracted much attention and has been involved in almost every area of computer science. Since the first original work by Yang in 2010, many works about BA have been published.

Continuous optimization:
Extensive studies have applied BA for continuous optimization problems. These studies showed that Bat Algorithm can efficiently handle nonlinear problems and easily obtain the optimal solutions in engineering design. For example, BA has been applied by Tsai et al. [77] to solve numerical optimization problems. In addition, it has been used by Bora et al. [10] Bora et al. [10] for the brushless DC wheel motor problem.

Combinatorial optimization and scheduling:
Although continuous optimization problems are still quite difficult to solve, they can be easily solved in terms of computational cost. In contrast, solving combinatorial optimization problems can be very challenging and hard in terms of computational complexity. A detailed research of combined economic load and emission dispatch using Bat Algorithm has been introduced by Ramesh et al. [59]. BA has been also used in the work of Musikapun and Pongcharoen [56] in order to solve multi-stage, multi-machine, multi-product scheduling problems.

Inverse problems and parameter estimation:
Inverse problems and parameter estimation are another important area of applications of BA. For example, Bat Algorithm was used by Yang et al. [86] in order to topological shape optimization in microelectronic applications. On the other hand, BA can be involved to perform parameter estimation as an inverse problem. However, in order that BA can achieve better outcomes, the inverse problem should be expressed properly.

Classifications, clustering, and data mining:
In the context of e-learning, a comparison study of BA with other evolutionary and swarm algorithms has been discussed by Khan and Sahari [40]. This study showed that BA achieved better results over the other algorithms. Khan et al. [41] also involved BA for b-sonar problems. This work studied clustering problems using BA and yielded significant outcomes. On the other hand, a clustering using BA has been introduced by Komarasamy and Wahi [43]. In this work, BA was combined with K-means clustering. The experimental results showed that the proposed algorithm succeeded to achieve higher efficiency and obtained significant improvement over the other algorithms.

Image Processing:
A variant of BA for image matching was introduced by Du and Liu [89]. In this work, the authors mentioned that the proposed BA-based model is more effective than others models for image matching.

Fuzzy Logic:
A detailed study of fuzzy systems using BA has been presented by Tamiru and Hashim [75] in order to model energy changes in a gas turbine. In distribution systems, a study of optimal capacitor placement for loss reduction using BA has been discussed by Reddy and Manoj [60]. In this study, the authors combined BA and Fuzzy logic to obtain optimal capacitor placement

3.2.2 Firefly Algorithm

Firefly Algorithm is based on the flashing behavior of fireflies and the phenomenon of bioluminescent communication. Fireflies, also referred to as lightening bugs, are luminous beetles from the family Lampyridae. Most of fireflies are characterized by their use of bioluminescence to communicate, attract mate, hunt for prey, and to warn predators of their bitter taste. There are up to 2000 different species of fireflies, found in temperate and tropical environments around the world. All of them fall under five main subfamilies: Photurinae, Luciolinae, Cyphonocerinae, Lampyrinae, and Otetrinae.

a. Standard Firefly Algorithm

The FA procedure is based on interactions between individual fireflies. The level of interaction is modeled by the strength of this event. Each firefly has its attractiveness, which is used to attract other fireflies to it. Attractiveness depends on the light intensity, so each firefly is attracted to the neighbor that glows brighter. Relying on the behavior of the flashing characteristics of fireflies, the Firefly Algorithm was designed with the following three rules:

- All fireflies are unisex so that one firefly is attracted to other fireflies regardless of their sex.
- The attractiveness of a firefly is proportional to its light intensity which decreases as the distance from the other firefly increases. If there is not a more attractive firefly than a particular one, it will move randomly.
- The light intensity of a firefly is affected or determined by the landscape of the objective function. For maximization problems, the light intensity is proportional to the value of the fitness function.

b. Firefly movement

The FA is a population-based algorithm, where each population member (i.e., firefly) represents a candidate solution of the problem to be solved and thus denotes a point in the search space.

In simulation, at iteration t, each firefly i in the population is characterized by a position x_i^t. The position of each solution is evaluated by calculating its fitness value $f(x_i)$. The overall best solution x_* is selected according to the light intensity I_i, which is expressed by the objective function $f(x_i)$. The position x_i^{t+1} of each member in the swarm is adjusted by moving firefly i to another more attractive firefly j according to Equation 3.7:

$$x_i^{t+1} = x_i^t + \beta(x_j^t - x_j^t) + \alpha \left(rand - \frac{1}{2} \right) \tag{3.7}$$

Equation 3.7 consists of three terms. The first term determines the current position of the ith firefly. The second term refers to a social component of moving the firefly i towards the more attractive firefly j, while the third term is connected with the randomized move of the ith firefly within the search space.

In the second term of Equation 3.7, parameter β is the attractiveness of firefly j. It is described by monotonically decreasing function of the distance r_{ij} between the fireflies i and j, as shown in Equation 3.8:

$$\beta = \beta_0 e^{-\gamma r_{ij}^2} \tag{3.8}$$

Where:

β_0, is the maximum attractiveness value (i.e. at $r = 0$);

γ, is the light absorption coefficient, which controls the decrease of the light intensity;

r_{ij}, is the distance between the fireflies i and j at the positions x_i and x_j. It is obtained by the Cartesian distance as defined in Formula 3.9:

$$r_{ij} = \| x_i - x_j \| = \sqrt{\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2} \tag{3.9}$$

Where:

$x_{i,k}$, is the kth element of the ith firefly position within the search space;

d, the dimensionality of a problem.

In the third term of Equation 3.7, α is a randomization parameter and $rand$ is a random number generator uniformly distributed in the range $[0, 1]$.

c. Controlling randomization

To prevent the algorithm from the premature convergence, the randomization parameter α can be varied and gradually decreased using Formula 3.10:

$$\alpha = \alpha_0 \theta^t \tag{3.10}$$

Where:

α_0, is the initial randomization parameter;

θ, is the randomness reduction constant;

$t \in [0, T]$, is the pseudo time for simulations and T is the maximum number of generations.

d. The original Firefly Algorithm

The main steps of the Firefly Algorithm are presented in Algorithm 2 and its description can be summarized as follows:

Algorithm 2 pseudo code of the Firefly Algorithm (FA)

```
Fitness function f(x)
Define light absorption coefficient γ, Initial attractiveness β_0, parameter α
Light intensity I_i at x_i is determined by f(x_i)
Generate an initial population of N fireflies x_i (i = 1, 2, ..., N)
while (t < Max number of iterations) do
    for i ← 1 to N (all N fireflies) do
        for j ← 1 to N (all N fireflies) do
            if I_i < I_j then
                Move firefly i towards j (Equation 3.7);
            end if;
            Vary attractiveness;
            Evaluate new solutions and update light intensity;
        end for
    end for
    Rank the fireflies and find the current global best x_*;
end while
```

The algorithm starts by setting the initial values of the maximum attractiveness β_0, the absorption coefficient γ, and randomization parameter α. The initial population of N solutions $(i = 1, 2, ..., N)$ is generated randomly, where each solution is initialized with a location x_i^0. The light intensity I_i^0 of each solution in the initial population is determined by the fitness function value $f(x_i^0)$. The best solution x_* is then initialized by the best solutions of all the fireflies. The FA iteratively moves from the initial set of solutions to the best solution by adjusting the position x_i of firefly i to the position x_j of another more attractive firefly j according to Equation 3.7. Solutions are continuously updated based on the continuous flying iterations. This is repeated till the termination criteria are satisfied. Finally, when all criteria successfully met the best so far solution is reported.

e. Applications

Since the original Firefly Algorithm was instroduced by Yang in 2008, FAs have been adapted in almost every field of optimization, classifications, image processing, feature selection, scheduling, data mining, and other problems.

Continuous optimization:
Kazemzadeh [36] and Gandomi et al. [19] showed that Firefly Algorithm can efficiently fix multimodal design and nonlinear issues in engineering design optimization. Imanirad et al. [27] applied FA to produce alternatives for decision makers with diverse options. Moreover, FA has been used in antenna design optimization[13] [6].

Scheduling and Combinatorial optimization:
Scheduling and Combinatorial optimization are another important area of applications of FA with great performance. For instance, A discrete version of FA was introduced by Sayadi et al. [68] to efficiently solve NP-hard scheduling problems. Furthermore, FA was proposed with the aim to efficiently solve scheduling and traveling salesman problems [31, 88]. For example, a Firefly

Algorithm was proposed by Yousif et al. [88] in order to solve scheduling jobs on grid computing, whereas a discrete Firefly Algorithm was developed by Jati et al. [31] in order to fix the popular traveling salesman issue. FA also demonstrated its efficiency over a wide range of problems such as load dispatch problems. Swarnkar [74] used FA to solve economic load dispatch problems with reduced power losses. Similarly, Yang et al. [85] solved the non-convex economic dispatch problem with valve-loading effect using FA and obtained the best outputs compared to the other methods.

Classifications, clustering, and data mining:
Senthilnath et al. [69] performed extensive experiments study by comparing FA with several algorithms and showed that FA can efficiently used for Clustering. In addition, Tang et al. [76] provided a comprehensive review of nature-inspired algorithms for clustering.

Image Processing:
Banati and Bajaj [5] applied FA for feature selection. In this work, the authors demonstrated that Firefly Algorithm achieved better performance over others methods in terms of CPU time. In the same sense, Zhang and Wu [91] used FA to study image registration, whereas Horng et al. [26] showed that the proposed approach based on FA for digital image compression provided the least computation time.

3.2.3 Particle swarm optimization

One of the most efficient and widely used swarm intelligence algorithms is Particle Swarm Optimization (PSO) proposed by Kennedy and Eberhart in 1995 [37]. PSO is inspired by the social foraging behavior of some animals such as bird flocking and fish schooling behavior. It uses real-numbers randomness and global communication among the swarm particles rather than use the mutation, crossover or pheromone. Due to its simplicity and attractive search

efficiency, PSO has been successfully applied to various domains such as: combinatorial optimization [23], data mining [55], clustering [65], neural networks [61], image processing [32], scheduling [90], and fuzzy logic [50].

a. The standard PSO

The PSO is a population-based algorithm that consists of a set of candidate solutions called swarm particles. Each solution or swarm particle is characterized by a set of parameters and represents a point in multidimensional space. Particles move across the search space with a specified velocity and adjust their positions in the search space to explore higher fitness positions by a number of repeated iterations and finally to bring them to the position of the highest fitness value. Each particle accelerates in the direction of its own personal best solution found so far, as well as in the direction of the global best position discovered so far by any of the particles in the swarm. The velocities of particles are adjusted according to their own memory of the past experiences and that of the neighbors, while they fly through the search space. Each move of particles is deeply influenced by its current position, its memory of previous useful parameters, and the group knowledge of the swarm.

To model the swarm, at iteration t, each particle i is associated with a position x_i^t and a velocity v_i^t. The position of each solution is evaluated by calculating its fitness value $f(x_i)$ and the particle with the highest fitness value is considered as the global best solution x_*. Using its new velocity v_i^{t+1}, the position x_i^t of each particle i is updated according to Equation 3.11:

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{3.11}$$

The velocity v_i^{t+1} is determined according to the best position p_i that particle i has so far visited and the global best x_* found by all particles so far. Hence, the new velocity is given by Formula 3.12:

$$v_i^{t+1} = \theta v_i^t + \alpha\epsilon_1(x_* - x_i^t) + \beta\epsilon_2(p_i - x_i^t) \tag{3.12}$$

Where:

θ, is the inertia weight that takes values between 0 and 1;

α, β, are the learning parameters;

$\epsilon_1, \epsilon_2 \in [0, 1]$, are random vectors drawn from a uniform distribution.

b. The Accelerated PSO

The standard PSO uses both the personal best position p_i and the current global best x_* to update the velocity of particles. The usage of the personal best position is primarily to increase the diversity of the swarm; however, this diversity can be achieved using some randomness. Therefore, there is no compelling reason for using the personal best position. To accelerate the convergence of the algorithm, a simplified version of PSO called an accelerated particle swarm optimization (APSO) was proposed by yang in 2008 [81], in which only the global best solution is involved. The velocity v_i^{t+1} in APSO is given by Equation 3.13:

$$v_i^{t+1} = v_i^t + \beta(x_* - x_i^t) + \alpha(\epsilon - \frac{1}{2}) \tag{3.13}$$

The APSO can be further improved by formulating the update of particle's position in a single step as mentioned in Equation 3.14:

$$x_i^{t+1} = x_i^t + \beta(x_* - x_i^t) + \alpha(\epsilon - \frac{1}{2}) \tag{3.14}$$

Equation 3.14 consists of three terms. The first term determines the current position of the ith particle. The second term refers to a social component of moving the particle i toward the position of the current global best particle, while the third term is connected with the randomized move of the ith particle within the search space.

In the second term of Equation 3.14, the parameter β expresses the attractiveness of the global best solution. Its value can be increased stepwise from 0 to 1 to speed up the convergence of APSO [20].

In the third term of Equation 3.14, α is a randomization parameter and ϵ is a random number generator uniformly distributed in the range $[0, 1]$. To prevent

the algorithm from the premature convergence, the randomization parameter α can be varied and gradually decreased using Formula 3.15:

$$\alpha = \alpha_0 \gamma^t \tag{3.15}$$

Where:

α_0, is the initial randomization parameter;

γ, is the control parameter;

$t \in [0, T]$, is the pseudo time for simulations and T is the maximum number of generations.

c. The APSO Algorithm

The main steps of the accelerated particle swarm optimization are presented in Algorithm 3 and its description can be summarized as follows:

Algorithm 3 Pseudo code of the Accelerated Particle Swarm Optimization

```
Fitness function f(x)
Define the attraction parameter β, Randomization parameter α
Generate an initial population of N particles x_i (i = 1, 2, ..., N)
while (t < Max number of iterations) do
    for i ← 1 to N do
        Calculate new locations x^{t+1} (Equation 3.14);
        Evaluate fitness functions at new locations x^{t+1};
    end for
    Rank the particles and find the current global best x_*;
end while
```

The algorithm starts by setting the initial values of the attraction parameter β and randomization parameter α. The initial population of N particles $(i = 1, 2, ..., N)$ is then generated randomly, where each solution is initialized with a location x_i^0. The initial swarm is evaluated by calculating the fitness function value $f(x_i^0)$ of each solution. The best solution x_* is then initialized by the best solutions of all the particles. The APSO iteratively moves from the initial set of solutions to the best solution by updating the position x_i of each particle i using Equation 3.14. Solutions are continuously adjusted based on the continuous flying iterations. This is repeated till the termination criteria are

satisfied. Finally, when all criteria successfully met the best so far particle x_* is reported.

Chapter 4

Swarm Intelligence for Query Expansion in Web IR

In this chapter, we present our proposal for query expansion in Web information retrieval. First, we discuss our earlier approach to expand the original user query with extra terms that best express the actual user intent (Section 4.1). In this approach, we introduce two criteria to assess the degree of relatedness between a candidate expansion term and the search query keywords: (1) The co-occurrence, which uses the Good Turing Discounting (GTD) to attribute more importance to words that occur in the largest possible number of documents where the search query keywords appear; (2) The proximity and closeness, which use the Kernel functions to assign more importance to words that have a short distance with the query terms within the documents. In addition, we adopt the strength Pareto fitness assignment to satisfy both criteria simultaneously.

In the second part of this chapter (Section 4.2), we propose to consider the problem of query expansion as a combinatorial optimization problem and address it with swarm intelligence. The overall procedure consists in, first selecting the pseudo-relevant documents to the initial query, extract their keywords to build potential solutions and finally launch the swarm intelligence algorithm to determine the best expanded query.

4.1 Pareto dominance for selecting expansion keywords

The main goal of the proposed approach is to improve the retrieval effectiveness and return only the documents that are relevant to the search query. For that purpose, in this paper, we introduce the concepts of co-occurrence and closeness to extract the best expansion keywords to be added to the original search query. These concepts are based, at first, on finding for each query term Q_i the locations and positions where it appears and then selecting from these locations the candidate terms which frequently neighbor and co-occur with that query term. In other words, we recover for each query term Q_i the documents where it appears, and then assess the relevance of the candidate terms contained in these documents with respect to the query term Q_i on the basis of:

- The co-occurrence, which gives value to candidate words that appear in the largest possible number of those documents.

- The proximity and closeness, which gives value to candidate words in which the distance separating them and the query term Q_i within a given document, in terms of the number of words, is small.

These candidate words are then sorted on the basis of their relevance to the whole query and the top ranked ones are added to that query at the aim to repeat the search process and get more relevant results.

Before proceeding to describe the concepts of co-occurrence and proximity, we first need to represent each candidate term $t_i \in \grave{V}$ by a vector T_i of $|R|$ elements, as follows:

$$T_i =< pos_1, pos_2, ..., pos_{|R|} > \tag{4.1}$$

Where:

R, is the set of pseudo-relevant documents;

$\grave{V}$, is the vocabulary of R;

pos_k, is the position(s) of t_i in the pseudo-relevant document d_k. In the case where $t_i \notin d_k$, the value of pos_k will be 0; otherwise, its value will be a vector

containing all possible positions of t_i in d_k.

As indicated earlier, as a first step, we find the candidate words which often appear together with the query keywords. Finding these candidate words is performed by attributing more value to terms that occur in the largest possible number of documents where each of the query keywords appears. We interpret this value through the measurement of the external correlation ext of each term $t_i \in \grave{V}$ to each term $t_{j(Q)}$ of the query Q. This correlation, which does not take into consideration the content of documents, computes the rate of appearance of t_i with $t_{j(Q)}$ in the set of documents R. The external correlation of t_i to $t_{j(Q)}$ is significant when t_i appears in the largest number of documents in which $t_{j(Q)}$ occurs, and vice versa. Based on this interpretation, the external correlation of t_i to $t_{j(Q)}$ is calculated using the good Turing discounting method, as given by Equation 4.2:

$$ext\left(t_i, t_{j(Q)}\right) = \frac{1}{C(t_{j(Q)})}\left[\left(C(t_i, t_{j(Q)}) + 1\right)\frac{N_{C+1}}{N_C}\right] \tag{4.2}$$

Where:

$C(t_{j(Q)})$, is the number of times that $T_{j(Q)}[k] \neq 0$, $k = 1, ..., |R|$ (i.e., the number of documents where $t_{j(Q)}$ occurs in R);

$C(t_i, t_{j(Q)})$, is the number of times that $(T_{j(Q)}[k], T_i[k]) \neq 0$, $k = 1, ..., |R|$ (i.e., the number of documents where t_i and $t_{j(Q)}$ occur together in R);

N_{C+1}, is the number of pairs of terms that include $t_{j(Q)}$ and occur $C+1$ times in R;

N_C, is the number of pairs of terms that include $t_{j(Q)}$ and occur C times in R.

The Good Turing Discounting (GTD) has been widely used for computing the probability of a complete string of words or giving probabilistic prediction of what the next word will be in a sentence. Practically, the GTD has been involved to assign a non-zero probability to sequences of N words (N-grams) with zero or low counts by looking at the number of N-grams with higher counts [34]. Our dependence on good Turing discounting comes to solve the

issue that we faced in our previous works. In those works, we attempted to adopt the classical conditional probability to compute the rate of appearance of a given term to another one. The main problem of using the conditional probability was that the words, originally with a low frequency of occurrence, were neglected. Thus, their overall rates of appearance were automatically decreased. Accordingly, to avoid dropping the candidate words with low frequency, we use good Turing discounting to re-estimate their low-probabilities and improve their low appearance rates.

After computing the rate of appearance of t_i with each query term $t_{j(Q)}$, the overall external correlation between t_i and the whole query Q is represented by an array containing all possible ext between t_i and each query term $t_{j(Q)}$:

$$ext(t_i, Q) =< ext(t_i, t_{1(Q)}), ext(t_i, t_{2(Q)}), ..., ext(t_i, t_{|Q|(Q)}) > \tag{4.3}$$

The cosine similarity measure is then used to evaluate the quality of each vector $ext(t_i, Q)$ with respect to the best vector $ext(t_i^*, Q)$, where each of its elements $ext(t_i^*, t_{j(Q)})$ represents the highest external correlation between the term t_i and $t_{j(Q)}$. The following function, $f_{ext}(t_i)$, indicates the cosine similarity score between $ext(t_i^*, Q)$ and $ext(t_i, Q)$:

$$f_{ext}(t_i) = \frac{\sum_{j=1}^{|Q|} ext(t_i, t_{j(Q)}) * ext(t_i^*, t_{j(Q)})}{\sqrt{\sum_{j=1}^{|Q|} \left[ext(t_i, t_{j(Q)})\right]^2} * \sqrt{\sum_{j=1}^{|Q|} \left[ext(t_i^*, t_{j(Q)})\right]^2}} \tag{4.4}$$

In the second step, we try to find the candidate words which are often neighbors to the query keywords. Therefore, we assign more value to words having a short distance with the query keywords. We interpret this importance through the measurement of the internal correlation between each term t_i of $\grave{V}$ and each term $t_{j(Q)}$ of the query Q. This correlation computes the correlation between t_i and $t_{j(Q)}$ within a given document d_k in terms of the number of words separating them. The more t_i is close to $t_{j(Q)}$ within d_k, the greater is its internal correlation. We use the well-known Kernel functions to measure the internal correlation:

Gaussian kernel:

$$K(i,j) = \exp\left[\frac{-(i-j)^2}{2\sigma^2}\right] \tag{4.5}$$

Triangle kernel:

$$K(i,j) = \begin{cases} 1 - \frac{i-j}{\sigma}, & \text{if } |i-j| <= \sigma \\ 0, & \text{otherwise} \end{cases} \tag{4.6}$$

Cosine kernel:

$$K(i,j) = \begin{cases} \frac{1}{2}\left[1 + \cos\left(\frac{|i-j|\pi}{\sigma}\right)\right], & \text{if } |i-j| <= \sigma \\ 0, & \text{otherwise} \end{cases} \tag{4.7}$$

Where:

σ, is a parameter to be tuned.

Using the Kernel functions, the internal correlation int between t_i and $t_{j(Q)}$ within a given document d_k is then calculated by Equation 4.8:

$$int(t_i, t_{j(Q)})_{d_k} = K\left(T_i[k], T_{j(Q)}[k]\right) \tag{4.8}$$

Next, the average internal correlation between t_i and $t_{j(Q)}$ in the whole R is determined as follows:

$$int(t_i, t_{j(Q)}) = \frac{1}{C(t_{j(Q)})} \sum_{d_k \in R} int(t_i, t_{j(Q)})_{d_k} \tag{4.9}$$

The overall internal correlation between t_i and the whole query Q is described by a vector containing all possible int between t_i and each query term $t_{j(Q)}$:

$$int(t_i, Q) =< int(t_i, t_{1(Q)}), int(t_i, t_{2(Q)}), ..., int(t_i, t_{|Q|(Q)}) > \tag{4.10}$$

The cosine similarity measure is then used to assess the quality of each array $int(t_i, Q)$ with respect to the best vector $int(t_i^*, Q)$ where each of its elements $int(t_i^*, t_{j(Q)})$ represents the highest internal correlation between a given term t_i and $t_{j(Q)}$. The following function, $f_{int}(t_i)$, indicates the cosine similarity score

between $int(t_i^*, Q)$ and $int(t_i, Q)$:

$$f_{int}(t_i) = \frac{\sum_{j=1}^{|Q|} int(t_i, t_{j(Q)}) * int(t_i^*, t_{j(Q)})}{\sqrt{\sum_{j=1}^{|Q|} \left[int(t_i, t_{j(Q)})\right]^2} * \sqrt{\sum_{j=1}^{|Q|} \left[int(t_i^*, t_{j(Q)})\right]^2}} \tag{4.11}$$

Finally, in order to extract the suitable terms to be used as expansion keywords, we adopt the well-known concept of Pareto dominance. Thus, instead of using the conventional methods in determining the overall correlation such as summing the internal and external correlations or adjusting the balance between them, we consider both internal and external correlations as multiple conflict criteria to be fulfilled simultaneously. The concept of Pareto dominance was proposed to solve the multi-objective optimization problem, also called multi criteria optimization. The multi-objective optimization problem can be defined as the problem of finding such a solution which satisfies an objective vector whose elements represent the objectives functions. The solution to this problem can be described in terms of a decision vector $(x_1, x_2, x_3, ..., x_n)$ in the decision space X. A function $f : X \rightarrow Y$ evaluates the quality of a given solution by assigning it an objective vector $(y_1, y_2, y_3, ..., y_k)$ in the objective space Y. We say that a decision vector x^1 is better than another decision vector $x^2 (x^1 > x^2)$ if the objective vector y^1 dominates the objective vector $y^2 (y^1 > y^2)$ where $y^1 = f(x^1)$, $y^2 = f(x^2)$ and $k > 1$. The vector y^1 is said to dominate the vector y^2 if no component of y^1 is smaller than the corresponding component of y^2, and at least one component of y^1 is greater than the corresponding component of y^2. The set of optimal solutions, i.e., solutions not dominated by any other solutions, in the decision space X is denoted as the Pareto set $X^* \subseteq X$ and its image in objective space is denoted as Pareto front $Y^* = f(X^*) \subseteq Y$. Many enhanced functions have been proposed and extended to describe the Pareto dominance concept. One of the most improved dominance functions is the strength Pareto fitness assignment (SPEA2). It assigns each solution x^i a strength value $S(x^i)$ representing the number of solutions it dominates, as given by Equation 4.12:

$$S(x^i) = |x^j | x^j \in X \wedge x^i > x^j| \tag{4.12}$$

Where:

$|.|$, is the cardinality of set;

$>$, is the Pareto dominance relation;

$x^i > x^j$, if the objective vector y^i assigned to x^i dominates the objective vector y^j assigned to x^j.

On the basis of the S values, the raw fitness $R(x^i)$ of solution x^i is calculated by Equation 4.13:

$$R(x^i) = \sum_{x^j \in X, x^j > x^i} S(x^j) \tag{4.13}$$

It is important to note that the raw fitness R is to minimized here, i.e., $R(x^i) = 0$ corresponds to a non-dominated individual.

By analogy, for the proposed approach, a candidate expansion term $t_i \in \grave{V}$ is represented by a decision vector $T_i =< pos_1, pos_2, ..., pos_{|R|} >$. The quality of a given candidate term t_i is evaluated by assigning it an objective vector $f(t_i) =< f_{ext}(t_i), f_{int}(t_i) >$. We say that a candidate term t_i is better than another candidate term t_j $(t_i > t_j)$ if the objective vector $f(t_i)$ dominates the objective vector $f(t_j)$. The vector $f(t_i)$ is said to dominate the vector $f(t_j)$ if its components, $f_{ext}(t_i)$ and $f_{int}(t_i)$, are not smaller than their corresponding components in $f(t_j)$, and at least one component of $f(t_i)$ is greater than its corresponding component in $f(t_j)$. Based on the concept of dominance, each candidate term t_i is assigned a strength value $S(t_i)$ representing the number of expansion keyword candidates it dominates, as shown in Equation 4.14:

$$S(t_i) = |t_j | t_j \in \grave{V} \wedge t_i > t_j| \tag{4.14}$$

The raw fitness $R(t_i)$ of each candidate term $t_i \in \grave{V}$ is then calculated using Equation 4.15:

$$R(t_i) = \sum_{t_j \in \grave{V}, t_j > t_i} S(t_j) \tag{4.15}$$

Next, the candidate terms are sorted on the basis of their raw fitness values. The best candidate terms are the ones with the lowest raw fitness values and

the top ranked ones are added to the original search query Q.

Based on the Okapi BM25 document-scoring function, the relevant documents are retrieved using Equation 4.16:

$$Score_{BM25}(d, \grave{Q}) = \sum_{t_i \in Q} w_i^{BM25} + \frac{1}{2} \sum_{t_i \in \grave{Q} - Q} w_i^{BM25} * [f_{ext}(t_i) + f_{int}(t_i)] \quad (4.16)$$

Where:

$\grave{Q}$, is the expanded query.

4.2 Swarm intelligence for query expansion

In this section, we present a swarm intelligence-based approach to enhance the effectiveness of query expansion. In this approach, we first retrieve the pseudo-relevant documents, then we use SI algorithms to select the best expanded query among a set of candidates built from the pseudo-relevant documents, and finally we find the relevant documents to the best selected expanded query. Figure 4.1 shows the proposed SI algorithm for query expansion.

In the following, we present our proper modelling of query expansion with swarm intelligence. In particular, we describe the solution representation, the search space, the objective function and the parameters setting.

4.2.1 Solution representation

The choice of an efficient encoding for candidate solutions is one of the most important issues in designing a swarm intelligence algorithm. For the proposed approach, each candidate solution is encapsulated in a virtual element. The search space is then considered as a heart of artificial elements. A virtual element for our problem is an expanded query consisting of two parts: the first includes the words of the original query and the second is formed by other keywords belonging to R, the set of the pseudo-relevant documents retrieved in response to the original query. As the first part is fixed, only the words to be

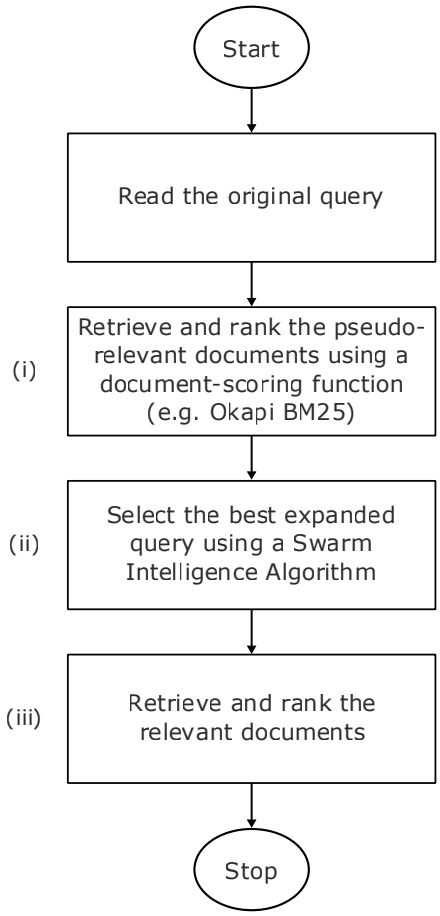


FIGURE 4.1: The proposed SI algorithm for Query expansion

added to the initial query are considered in representing the solution. Accordingly, the candidate solution is characterized by an array of words as given in 4.17:

$$\grave{Q} =< \grave{q}_1, \grave{q}_2, \grave{q}_3, ..., \grave{q}_{|\grave{Q}|} > \tag{4.17}$$

Where:

$\grave{Q}$, is a vector containing the terms to be added to the original search query. The overall expanded query is defined as follows:

$$Q + \grave{Q} =< q_1, q_2, ..., q_{|Q|}, \grave{q}_1, \grave{q}_2, ..., \grave{q}_{|\grave{Q}|} > \tag{4.18}$$

Where:

q_i, is a keyword belonging to the original query Q, $i = 1, ..., |Q|$;

$\grave{q}_i \in \grave{V}$, is a keyword to be added to the original query;

$\grave{V} \subset V$, is the vocabulary of R;

V, is the vocabulary of the whole collection;

$|Q|$, is the number of keywords in the original query;

$|\grave{Q}|$, is the number of expansion keywords.

The solution or search space contains all possible expanded parts without the fixed part of the expanded query. In other words, the search space includes all potential combinations of terms belonging to $\grave{V}$ having the same length $|\grave{Q}|$. The number of combinations of terms is given by the binomial coefficient, as indicated in Equation 4.19:

$$c(\grave{Q}, \grave{V}) = \frac{|\grave{V}|!}{|\grave{Q}|!(|\grave{V}| - |\grave{Q}|)!} \tag{4.19}$$

Where:

$c(\grave{Q}, \grave{V})$, is the number of combinations of $|\grave{Q}|$ terms among the total $|\grave{V}|$ terms.

This large number of combinations demonstrates the numerous possibilities to produce a query expansion. Considering the issue as an optimization problem, and addressing it with a proper approach is the optimal way to achieve effectiveness as well as efficiency.

4.2.2 Population initialization

The initial swarm of bats is generated with N elements. Each element in the initial swarm is a $|\grave{Q}|$-dimension vector, where each of its components is a keyword randomly selected from the set of terms $\grave{V}$.

4.2.3 Objective function

Each solution in the search space is associated with a numeric objective value. Hence, the quality of a solution is proportional to the value of the objective function (fitness). As a solution in our case is a combination of terms, the best

way to evaluate its performance is to take into consideration the inverted indexes of its keywords and calculate the scores of each document belonging to R with respect to the expanded query. The best score is then considered as the fitness value of the expanded query. The proposed fitness function is given by Formula 4.20 and summarized in Algorithm 4:

$$f(\grave{Q}) = max\left(Score_{BM25}(d_1, \grave{Q} + Q), ..., Score_{BM25}(d_{|R|}, \grave{Q} + Q)\right) \tag{4.20}$$

Algorithm 4 pseudo code of the objective function

```
Fitness function f(Q̀)
Inverted index I(q̀_i)
Initialize the vector of scores S ← []
for each keyword q̀_i in Q̀ do
    for each document d in R do
        if d ∈ I(q̀_i) and S[d] ≠ 0 then
            S[d] ← Score_BM25(d, Q) + Score_BM25(d, Q̀);
        end if;
    end for
end for
f(Q̀) ← max(S);
```

Algorithm 4 starts by retrieving the inverted index I of each query term $\grave{q}_i \in \grave{Q}$, and iteratively computes the score of each document d included in both R and I with respect to the query $\grave{Q}$. In order to avoid computing the same score many times, a check is made to ensure that no score has yet been attributed to document d in the vector of scores S. In addition, as the score of d to Q is calculated in the first step (i), only the score of d to $\grave{Q}$ is computed at this stage. Once all the scores are calculated, the maximum value of S is returned as the fitness value of the expanded query.

When the last generation of the proposed algorithm is achieved, the vector S of the best solution is sorted and its elements are considered as relevant documents to the user's original query.

4.2.4 Bat Algorithm: Update locations and velocities of bats

The searched solution changes all along the bat process in an increasing order of its fitness function. The current best solution of all the previous iterations is denoted by $\grave{Q}_*$.

At iteration t, each solution $\grave{Q}_i$ in the population has a position $\grave{Q}_i^t$ and a velocity v_i^t, which are both described by a vector of $|\grave{Q}|$ elements. The value of the kth element of the position vector $\grave{Q}_i^t$ is a keyword belonging to $\grave{V}$, whereas the value of the kth element of the velocity vector v_i^t can be 0 or a keyword belonging to $\grave{V}$. Based on Equation 3.1, the solution $\grave{Q}_i$ moves from the current position $\grave{Q}_i^t$ to the next position $\grave{Q}_i^{t+1}$ using its new velocity v_i^{t+1}, as follows:

$$\grave{Q}_i^{t+1} = \grave{Q}_i^t + v_i^{t+1} \tag{4.21}$$

The movement to the next position is expressed as an addition operation between the current position $\grave{Q}_i^t$ and the new velocity v_i^{t+1}. For the proposed BA for RF, the addition operation is evaluated as $X = X + V$ and the result is determined by an array of $|\grave{Q}|$ elements in which each kth element gets the value of $\grave{Q}_{i,k}^t$ if the element $v_{i,k}^{t+1}$ of v_i^{t+1} is equal to zero; otherwise, the kth element of the new position $\grave{Q}_i^{t+1}$ is set to $v_{i,k}^{t+1}$. The formula for position adjustment is defined by Equation 4.22:

$$\grave{Q}_{i,k}^{t+1} = \begin{cases} \grave{Q}_{i,k}^t, & \text{if } v_{i,k}^{t+1} = 0 \\ v_{i,k}^{t+1}, & \text{otherwise} \end{cases} \tag{4.22}$$

Regarding the new velocity v_i^{t+1}, it is given by Formula 4.23:

$$v_i^{t+1} = v_i^t + (\grave{Q}_i^t - \grave{Q}_*) f_i \tag{4.23}$$

To calculate the velocity v_i^{t+1}, we first need to compute the difference between the current position $\grave{Q}_i^t$ and the global best solution $\grave{Q}_*$. This difference is accomplished by the subtraction of two positions and evaluated as $V = X_1 - X_2$. The result of subtraction is presented by a vector of $|\grave{Q}|$ elements in which each kth element gets the value of $\grave{Q}_{*,k}$ if $\grave{Q}_{i,k}^t$ is different from $\grave{Q}_{*,k}$; otherwise, the

value of the kth element is set to 0. Equation 4.24 illustrates the manner in which the subtraction operation between $\grave{Q}_i^t$ and $\grave{Q}_*$ is applied:

$$\grave{Q}_{i,k}^t - \grave{Q}_{*,k} = \begin{cases} 0, & \text{if } \grave{Q}_{i,k}^t = \grave{Q}_{*,k} \\ \grave{Q}_{*,k}, & \text{otherwise} \end{cases} \tag{4.24}$$

Once the difference between $\grave{Q}_i^t$ and $\grave{Q}_*$ is evaluated, the result is multiplied by the frequency f_i. For the proposed Bat Algorithm, a random value $rand$ is generated. If $rand$ is greater than or equal to f_i, the corresponding element of the kth output is equal to 0; otherwise, it is set to $\grave{Q}_i^t - \grave{Q}_*$. The result of the multiplication operation is given by Formula 4.25:

$$(\grave{Q}_i^t - \grave{Q}_*)f_i = \begin{cases} 0, & \text{if } rand > f_i \\ \grave{Q}_i^t - \grave{Q}_*, & \text{otherwise} \end{cases} \tag{4.25}$$

Finally, the new velocity v_i^{t+1} is determined by computing the sum of the current velocity v_i^t and the output of Equation 4.25. This sum is accomplished by the addition of two velocities and evaluated as $V = V_1 + V_2$. The result of addition is presented by an array of $|\grave{Q}|$ elements in which each kth element gets the value of $v_{i,k}^t$ if a random number $rand$ is greater than 0.5; otherwise, it is set to the kth output of Equation 4.25. The new velocity is updated according to Equation 4.26:

$$v_{i,k}^{t+1} = \begin{cases} v_{i,k}^t, & \text{if } rand > 0.5 \\ (\grave{Q}_{i,k}^t - \grave{Q}_{*,k})f_i, & \text{otherwise} \end{cases} \tag{4.26}$$

In order to refine the best found solution, the new position $\grave{Q}_i^{t+1}$ is adjusted by generating a new solution through a random walk as given by Formula 4.27:

$$\grave{Q}_{i,k}^{t+1} = \begin{cases} \grave{Q}_{i,k}^{t+1}, & \text{if } rand > \epsilon A^{(t)} \\ \grave{q}_l, & \text{otherwise} \end{cases} \tag{4.27}$$

Where:

$\grave{q}_l$, is a keyword selected randomly from the set of terms $\grave{V}$.

For the proposed BA, the frequency f_{min} is set to 0 while the frequency f_{max}, the rate r and the loudness A are defined empirically. Another parameter to be experimentally estimated is the solution size $|\grave{Q}|$. These features are considered for the first time for query expansion approaches, and provide more strength to our proposal.

a. Pseudo code of the proposed Bat Algorithm for RF

The main steps of the proposed Bat Algorithm for retrieval feedback are summarized in Algorithm 5. The initial population of N solutions $(i = 1, 2, ..., N)$ is first created randomly. Each solution in the initial swarm is generated by selecting $|\grave{Q}|$ terms from $\grave{V}$ and initialized with a location $\grave{Q}_i^0$, a velocity v_i^0, and assigned an initial frequency f_i. The values of pulse rate r_i and loudness A_i are initialized and the initial population is then evaluated by calculating the objective function value of each solution $f(\grave{Q}_i^0)$. The BA iteratively moves from the initial set of expanded queries to the best expanded query by adjusting the position of $\grave{Q}_i$ and the velocity v_i as shown in Equations 4.21 and 4.23. The new solutions are evaluated and the best solution $\grave{Q}_*$ is selected. The local search method is applied by using a random walk method as given in Equation 4.27. The solutions are continuously updated based on the continuous flying iterations. This is repeated till the best solution does not change for a pre-specified interval of generations or the maximum number of iterations is reached. Finally, when the termination criteria are successfully met, the best so far expanded query is reported.

b. Computational complexity

The complexity of the proposed approach depends on both the complexity of the fitness function and the expected number of BA iterations. The computing complexity of the fitness function is in the order of $O(|R||\grave{Q}|N)$. Hence, the overall complexity of the algorithm is defined by $O(|R||\grave{Q}|NT)$.

Algorithm 5 pseudo code of the proposed Bat Algorithm for RF

Fitness function $f(\grave{Q})$
Initialize the bat population $\grave{Q}_i$ and v_i $(i = 1, 2, ..., N)$
Initialize frequencies f_i, pulse rates r_i and the loudness A_i
while ($t <$ Max number of iterations **and** the best expanded query still changes) **do**
    Generate new solutions by adjusting frequency (Equation 3.3);
    Update velocities and locations/solutions (Equations 4.21 and 4.23);
    if $rand > r_i$ **then**
        Select a solution among the best solutions;
        Generate a local solution around the selected best solution;
    end if;
    Generate a new solution by flying randomly;
    if $rand < A_i$ **and** $f(\grave{Q}_i) < f(\grave{Q}_*)$ **then**
        Accept the new solutions;
    end if;
    Rank the bats and find the current best expanded query;
end while

4.2.5 Firefly Algorithm: Update locations of fireflies

The searched solution changes all along the firefly process in an increasing order of its fitness function. The current best solution of all the previous iterations is denoted by $\grave{Q}_*$. At each iteration, the algorithm computes the best firefly of the generation, which becomes the global best one if it is better than $\grave{Q}_*$. To adapt an effective Firefly Algorithm for pseudo-relevance feedback issue, some modifications on the basic FA are made. Hence, the movement of firefly $\grave{Q}_i$ towards another more attractive $\grave{Q}_j$ is performed in two steps. The first step, which employs the second term of Equation 3.7, is expressed by Formula 4.28:

$$\beta = \frac{1}{1 + \gamma r_{ij}} \tag{4.28}$$

Where:

$\beta \in]0, 1]$, is the attractiveness of firefly j;

γ, is the light absorption coefficient;

r_{ij}, is the distance between $\grave{Q}_i$ and $\grave{Q}_j$. It is calculated using the well-known Hamming distance. The Hamming distance between two fireflies is determined by the number of non-corresponding elements in the sequence. For

example, given two solutions $\grave{Q}_1$ and $\grave{Q}_2$ containing 5 keywords:

$$\grave{Q}_1 =< \grave{q}_4, \grave{q}_2, \mathbf{\grave{q}_6}, \mathbf{\grave{q}_3}, \mathbf{\grave{q}_7} >$$
$$\grave{Q}_2 =< \mathbf{\grave{q}_1}, \grave{q}_4, \mathbf{\grave{q}_8}, \mathbf{\grave{q}_5}, \grave{q}_2 >$$

The Hamming Distance between $\grave{Q}_1$ and $\grave{Q}_1$ would be 3.

The importance of this step lies in bringing the solution $\grave{Q}_i$ closer to another more attractive solution $\grave{Q}_j$. In other words, after performing this step, the distance between $\grave{Q}_i$ and $\grave{Q}_j$ is decreased while the attractiveness value is increased. Moreover, the number of keywords of $\grave{Q}_j$ in $\grave{Q}_i$ at iteration $t+1$ is greater than or equal to this number at iteration t. Hence, the next position of $\grave{Q}_i$ is updated by moving each of its elements $\grave{Q}_{ik}$ (keyword), which does not belong to $\grave{Q}_j$, as shown in Equation 4.29:

$$\grave{Q}_{ik}^{t+1} = \begin{cases} \grave{Q}_{ik}^{t}, & \text{if } rand > \beta \\ \grave{Q}_{jl}^{t}, & \text{otherwise} \end{cases} \tag{4.29}$$

Where:

$\grave{Q}_{ik}^{t} \notin \grave{Q}_{j}^{t}$, is a keyword belonging to $\grave{Q}_{i}^{t}$;

$\grave{Q}_{jl}^{t} \notin \grave{Q}_{i}^{t}$, is a keyword selected randomly from $\grave{Q}_{j}^{t}$.

Once the first step is completed, the second stage is then conducted using the third term of Equation 3.7. For our problem, the third term is determined by the value of α. The randomization parameter α is very important in a Firefly Algorithm as it avoids the solutions being trapped at local optima. Its value is gradually decreased to prevent the algorithm from the premature convergence. Based on the value of α, the position of $\grave{Q}_i$ is updated by adjusting randomly each of its elements $\grave{Q}_{ik}$, which is not included in $\grave{Q}_j$, as given by Formula 4.30:

$$\grave{Q}_{ik}^{t+1} = \begin{cases} \grave{Q}_{ik}^{t}, & \text{if } rand > \alpha \\ \grave{q}_{l}, & \text{otherwise} \end{cases} \tag{4.30}$$

Where:

$\grave{q}_l$, is a keyword selected randomly from the set of terms $\grave{V}$.

Exploitation and exploration are two main features of any swarm algorithms. From the above two steps, it can be observed that those features are achieved. In the first step, the Exploitation is attained by focusing on the search in a local region by exploiting the information that a more attractive firefly is found in this region. In the second stage, the exploration is realized by exploring the search space and generating diverse solutions via the randomization. The good combination of these two major components will usually ensure that the global optimality is achievable.

a. Pseudo code of the proposed Firefly Algorithm for PRF

The main steps of the proposed Firefly Algorithm for pseudo-relevance feedback are summarized in Algorithm 6. The algorithm starts by setting the initial values of the absorption coefficient γ and the randomization parameter α. The initial population of N solutions $(i = 1, 2, ..., N)$ is then created randomly. Each solution in the initial swarm is generated by selecting $|\grave{Q}|$ terms from $\grave{V}$ and initialized with a location $\grave{Q}_i^0$. The light intensity I_i^0 of each solution in the initial population is determined by the fitness function value $f(\grave{Q}_i^0)$. The best solution $\grave{Q}_*$ is then initialized by the best solutions of all the fireflies. The FA iteratively moves from the initial set of solutions to the best solution by moving the firefly $\grave{Q}_i$ towards another more attractive $\grave{Q}_j$ using Equations 4.29 and 4.30. Solutions are continuously updated based on the continuous flying iterations. This is repeated till the best solution does not change for a pre-specified interval of generations or the maximum number of iterations is reached. Finally, when the termination criteria are successfully met, the best so far expanded query is reported.

Algorithm 6 pseudo code of the proposed Firefly Algorithm for PRF

```
Fitness function f(Q̀)
Define light absorption coefficient γ, parameter α
Light intensity I_i at Q̀_i is determined by f(Q̀_i)
Generate an initial population of N fireflies Q̀_i (i = 1, 2, ..., N)
while (t < Max number of iterations and the best expanded query still changes) do
    for i ← 1 to N (all N fireflies) do
        for j ← 1 to N (all N fireflies) do
            if f(Q̀_i) < f(Q̀_j) then
                Move query firefly Q̀_i towards Q̀_j (Equations 4.29 and 4.30);
            end if;
            Vary attractiveness;
            Evaluate new solutions;
        end for
    end for
    Rank the fireflies and find the current best expanded query Q̀_*;
end while
```

b. Computational complexity

The complexity of the proposed approach depends on both the complexity of the fitness function and the expected number of FA iterations. The computing complexity of the fitness function is in the order of $O(|R||\grave{Q}|N)$. Hence, the overall complexity of the algorithm is defined by $O(|R||\grave{Q}|NT)$

4.2.6 APSO: Update locations of particles

To adapt an effective APSO for query expansion issue, some modifications on the basic accelerated particle swarm optimization are made. Thus, the movement of particle $\grave{Q}_i$ towards the global best solution $\grave{Q}_*$ is performed in two steps. The first step, which employs the second term of Equation 3.13, is achieved by varying the value of attraction parameter β as given by Formula 4.31:

$$\beta = \frac{1}{1 + r_{i*}} \tag{4.31}$$

Where:

$\beta \in]0, 1]$, is the attractiveness of global best solution $\grave{Q}_*$;

r_{i*}, is the distance between $\grave{Q}_i$ and $\grave{Q}_*$. It is calculated using the well-known Hamming distance. The Hamming distance between two particles is determined by the number of non-corresponding elements in the sequence. For example, given two solutions $\grave{Q}_1$ and $\grave{Q}_2$ containing 5 keywords:

$$\grave{Q}_1 =< \grave{q}_4, \grave{q}_2, \mathbf{\grave{q}_6}, \mathbf{\grave{q}_3}, \mathbf{\grave{q}_7} >$$
$$\grave{Q}_2 =< \mathbf{\grave{q}_1}, \grave{q}_4, \mathbf{\grave{q}_8}, \mathbf{\grave{q}_5}, \grave{q}_2 >$$

The Hamming Distance between $\grave{Q}_1$ and $\grave{Q}_2$ would be 3.

The importance of this step lies in bringing the solution $\grave{Q}_i$ closer to the global best $\grave{Q}_*$. In other words, after performing this step, the distance between $\grave{Q}_i$ and $\grave{Q}_*$ is decreased while the value of attraction is increased. Moreover, the number of keywords of $\grave{Q}_*$ in $\grave{Q}_i$ at iteration $t+1$ is greater than or equal to this number at iteration t. Hence, the next position of $\grave{Q}_i$ is updated by moving each of its elements $\grave{Q}_{ik}$ (i.e. keyword), which does not belong to $\grave{Q}_*$, according to Equation 4.32:

$$\grave{Q}_{ik}^{t+1} = \begin{cases} \grave{Q}_{ik}^{t}, & \text{if } rand > \beta \\ \grave{Q}_{*l}^{t}, & \text{otherwise} \end{cases} \tag{4.32}$$

Where:

$\grave{Q}_{ik}^{t} \notin \grave{Q}_{*}^{t}$, is a keyword belonging to $\grave{Q}_{i}^{t}$;

$\grave{Q}_{*l}^{t} \notin \grave{Q}_{i}^{t}$, is a keyword selected randomly from $\grave{Q}_{*}^{t}$.

Once the first step is completed, the second stage is then conducted using the third term of Equation 3.13. For our problem, the third term is determined by the value of α. The randomization parameter α is very important in a APSO as it avoids the solutions being trapped at local optima. Its value is gradually decreased to prevent the algorithm from the premature convergence. Based on the value of α, the position of $\grave{Q}_i$ is updated by adjusting randomly each of its elements $\grave{Q}_{ik}^{t}$, which is not included in $\grave{Q}_*$, as given by Formula 4.33:

$$\grave{Q}_{ik}^{t+1} = \begin{cases} \grave{Q}_{ik}^{t}, & \text{if } rand > \alpha \\ \grave{q}_{l}, & \text{otherwise} \end{cases} \tag{4.33}$$

Where:

$\grave{q}_l$, is a keyword selected randomly from the set of terms $\grave{V}$.

Exploitation and exploration are two main features of any swarm algorithms. From the above two steps, it can be observed that those features are achieved. In the first step, the exploitation is attained by focusing on the search in a local region by exploiting the information that a current best solution is found in this region. In the second stage, the exploration is realized by exploring the search space and generating diverse solutions via the randomization. The good combination of these two major components will usually ensure that the global optimality is achievable.

a. Pseudo code of the proposed APSO for QE

The main steps of the proposed APSO for query expansion are summarized in Algorithm 7. The algorithm starts by setting the initial values of the attraction parameter β and randomization parameter α. The initial population of N solutions $(i = 1, 2, ..., N)$ is then created randomly. Each solution in the initial swarm is generated by selecting $|\grave{Q}|$ terms from $\grave{V}$ and initialized with a location $\grave{Q}_i^0$. The initial swarm is evaluated by calculating the fitness function value $f(\grave{Q}_i^0)$. The APSO iteratively moves from the initial set of expended queries to the best solution by updating the position $\grave{Q}_i$ of each particle i using Equations 4.32 and 4.33. Solutions are continuously adjusted based on the continuous flying iterations. This is repeated till the best solution does not change for a pre-specified interval of generations or the maximum number of iterations is reached. Finally, when the termination criteria are successfully met, the best so far expanded query reported.

Finally, as indicated earlier, the vector S of the best solution is sorted and its elements are considered as relevant documents to the user's original query.

Algorithm 7 Pseudo code of the proposed APSO for QE

Fitness function $f(\grave{Q})$
Define the attraction parameter β, Randomization parameter α
Generate an initial population of N particles $\grave{Q}_i$ $(i = 1, 2, ..., N)$
while ($t <$ Max number of iterations **and** the best expanded query still changes) **do**
    for $i \leftarrow 1$ **to** N **do**
        Calculate new locations $\grave{Q}^{t+1}$ (Equations 4.32 and 4.33);
        Evaluate fitness functions at new locations $\grave{Q}^{t+1}$;
    end for
    Rank the particles and find the current best expanded query;
end while

b. Computational complexity

The complexity of the proposed approach depends on both the complexity of the fitness function and the expected number of APSO iterations. The computing complexity of the fitness function is in the order of $O(|R||\grave{Q}|N)$. Hence, the overall complexity of the algorithm is defined by $O(|R||\grave{Q}|NT)$.

Chapter 5

Experimental results

In order to evaluate the performance of the proposed approaches for query expansion, we conduct extensive experiments on MEDLINE, the world Web's largest medical library. We first describe the dataset (section 5.1), the evaluation metrics (section 5.2) and the methods used for comparison (section 5.3). Then, we present the experimental results (section 5.4).

5.1 Dataset

Extensive experiments are performed on MEDLINE collection. This collection includes 348 566 references, consisting of titles and/or abstracts from 270 medical journals over a five-year period (1987-1991). The available fields are title, abstract, MeSH indexing terms, author, source, and publication type. In addition, the collection contains a set of queries, and relevance judgments (a list of which documents are relevant to each query). Regarding the queries, the MEDLINE collection includes 106 queries. Each query is accompanied by a set of relevance judgments chosen from the whole collection of documents. Table 5.1 summarizes the characteristics of the MEDLINE collection.

TABLE 5.1: Summary of MEDLINE collection and queries used in our experiments

Number of document in the collection	348566
Average document length in terms of words ($avdl$)	122
Number of words in the dictionary	308083
Number of queries	106
Average length query in terms of words	5.25

5.2 Evaluation Metrics

The precision (P) and the mean average precision (MAP) measures are used to evaluate the effectiveness of the proposed approaches. The CPU Time is used to assess the computational complexity of the proposed approaches.

5.3 Comparison with state-of-the-art methods

We compare the results of our proposal to the following well-known term-ranking functions, which based on analysis of term distribution in pseudo-relevant documents: Binary Independence Model (BIM), which is computed by Robertson & Sparck Jones weight, and Rocchio's weight.

5.4 Results

In this section, we first present the results of our proposal based on Pareto Fitness Assignment, and then we show the testing results of our proposed approaches based on swarm intelligence algorithms.

5.4.1 Pareto dominance for selecting expansion keywords

For this proposal approach, we split the MEDLINE dataset into 6 sub-collections. Each sub-collection is defined by a set of documents, queries, and a list of relevance documents. Table 5.2 highlights characteristics of each sub-collection in terms of the number of documents it contains (docs), the size of the sub-collection, and the number of words in the vocabulary.

TABLE 5.2: Summary of sub-collections used in our experiments.

Size of the **(#docs)**	**50000**	**100000**	**150000**	**200000**	**250000**	**300000**
collection (Mb)	26.39	52.36	80.72	107.58	135.05	164.31
Size of dictionary	81937	120825	156009	184514	211504	237889

The MEDLINE collection includes 106 queries. Each query is accompanied by a set of relevance judgments selected from the whole collection of

documents. Splitting the collection of documents into sub-collections leads inevitably to a decrease in the number of relevant documents for each query. In other words, if we have n relevant documents to a given query q with respect to the whole collection, then we will certainly have m relevant documents to the same query with respect to one of the sub-collections, where the value of n is greater or equal to the value of m. Furthermore, the probability of absence of any relevant document to a given query is not zero. In this case, each query does not include any relevant document in a given sub-collection is removed. Table 5.3 presents, for each sub-collection, the number of queries (Nb Queries), the average query length in terms of number of words (Avr Query Len) and the average number of relevant documents (Avr Rel Doc).

TABLE 5.3: Statistics on the MEDLINE sub-collections queries.

(#docs)	**50000**	**100000**	**150000**	**200000**	**250000**	**300000**
Nb Queries	82	91	95	97	99	101
Avr Rel Doc	4.23	7	10.94	13.78	15.5	19.24
Avr Query Len	6.79	6.12	5.68	5.74	5.62	5.51

Before proceeding to evaluate the performance of the proposed approach, we first fix the parameter σ of the Kernel functions that are used to compute the internal correlation. For that purpose, as a preliminary experiment, we consider the internal correlation as the overall correlation and systematically test a set of fixed σ values from 1 to 40 in increments of 5. Table 5.4 shows the precision values after retrieving 5 documents ($P@5$) and the mean average precision achieved, while using the sub-collection of 50000 documents. The number of pseudo-relevant documents R is tuned at 10, 20 and 50, and the number of expansion keywords is set to 10, which is the typical choice according to Carpineto and Romano [12]. Carpineto and Romano also demonstrated that this number can be increased to 30 keywords. Therefore, in case we have candidate keywords that have the same score, we do not attempt to distinguish them and simply add all to the original search query.

As it may be seen from Table 5.4, the suitable values of σ, which bring the highest performance, are 5 (10 out of 18), 10 (9 out of 18), 30 (7 out of 18) and 25

TABLE 5.4: The best performance of the proposed approach for different σ.

Kernel	R	σ	1	5	10	15	20	25	30	35	40
Gaussian	*10*	*P@5*	0.1560	**0.1682**	**0.1682**	0.1658	0.1658	0.1658	0.1658	0.1658	0.1658
		MAP	0.2207	0.2253	**0.2265**	0.2231	0.2230	0.2230	0.2230	0.2230	0.2230
	20	*P@5*	0.1609	**0.1682**	**0.1682**	**0.1682**	**0.1682**	**0.1682**	**0.1682**	**0.1682**	**0.1682**
		MAP	0.2208	0.2252	**0.2255**	0.2245	0.2245	0.2245	0.2245	0.2245	0.2245
	50	*P@5*	0.1609	**0.1682**	0.1658	0.1658	**0.1682**	**0.1682**	**0.1682**	**0.1682**	**0.1682**
		MAP	0.2193	**0.2241**	0.2231	0.2228	0.2235	0.2233	0.2233	0.2233	0.2233
Triangle	*10*	*P@5*	0.1609	**0.1707**	0.1682	0.1634	0.1682	0.1682	0.1682	0.1682	0.1682
		MAP	0.2110	0.2245	0.2234	0.2250	0.2257	0.2258	**0.2273**	0.2271	0.2252
	20	*P@5*	0.1609	**0.1731**	0.1682	0.1658	0.1682	0.1682	0.1682	0.1658	0.1658
		MAP	0.2110	0.2235	0.2211	0.2234	0.2249	0.2265	**0.2274**	0.2271	0.2252
	50	*P@5*	0.1609	**0.1682**	**0.1682**	0.1658	**0.1682**	**0.1682**	0.1658	0.1634	0.1634
		MAP	0.2110	0.2200	0.2200	0.2235	0.2248	0.2252	**0.2259**	0.2255	0.2235
Cosine	*10*	*P@5*	0.1609	**0.1682**	**0.1682**	0.1658	**0.1682**	**0.1682**	**0.1682**	**0.1682**	0.1658
		MAP	0.2110	0.2248	0.2255	0.2249	0.2264	0.2261	**0.2278**	0.2255	0.2232
	20	*P@5*	0.1609	**0.1707**	**0.1707**	0.1682	0.1658	0.1658	0.1658	0.1682	0.1682
		MAP	0.2110	0.2239	0.2229	0.2251	0.2255	**0.2267**	0.2251	0.2255	0.2247
	50	*P@5*	0.1609	**0.1682**	**0.1682**	0.1658	0.1658	0.1658	0.1658	0.1658	0.1658
		MAP	0.2110	0.2253	**0.2265**	0.2231	0.2230	0.2230	0.2230	0.2230	0.2230

(5 out of 18). In terms of MAP, the best results are achieved for $\sigma = 30$ in four cases, $\sigma = 10$ in three cases, $\sigma = 5$ in one case and $\sigma = 25$ in one case.

In the first phase of comparison, we evaluate the effectiveness of our proposed method through the use of only the external correlation, only the internal correlation, and both the external and internal correlations. In this experiment, the parameter σ is set to 5 and both the pseudo-relevant documents and the expansion keywords are set to 10. Tables 5.5(a) and 5.5(b) present, for each sub-collection, the precision values obtained by the external correlation (EXT), the internal correlation (INT), and both the external and internal correlations (EXT/INT) after retrieving 5 and 10 documents. In Table 5.5, *Rate* indicates the percentage of precision improvement of EXT/INT over EXT and INT.

Through Table 5.5(a), we can clearly see that EXT/INT produces the highest $P@5$ values for all sub-collections and achieves highly significant improvement over EXT and INT (Gaussian, Triangle, Cosine), e.g. on the 200000 sub-collection, there is an improvement (by EXT/INT (Triangle)) of 13.35% over EXT, 13.35% over INT (Gaussian), 10.20% over INT (Triangle) and 9.17% over INT (Cosine). Similarly, relevance precision at 10 retrieved documents improves from 0.1835 ($+7.85\%$), 0.1824 ($+8.50\%$), 0.1824 ($+8.50\%$) and 0.1835 ($+7.85\%$)

TABLE 5.5: Comparing the performance of EXT/INT, EXT and INT approaches in terms of precision.

(a) Precision after retrieving 5 documents *(P@5)*.

#docs	P	EXT/INT			EXT	INT		
		Gaussian	*Triangle*	*Cosine*		*Gaussian*	*Triangle*	*Cosine*
100000	*P@5*	0.1979	0.1845	0.1846	0.1626	0.1758	0.1736	0.1692
	Gaussian				**+21.65%**	+12.51%	**+13.94%**	**+16.90%**
	Rate		*Triangle*		+13.47%	+4.95%	+6.28%	+9.04%
				Cosine	+13.53%	+5.01%	+6.34%	+9.10%
200000	*P@5*	0.2432	0.2453	0.2432	0.2164	0.2164	0.2226	0.2247
	Gaussian				+12.38%	+12.38%	+9.25%	+8.23%
	Rate		*Triangle*		+13.35%	**+13.35%**	+10.20%	+9.17%
				Cosine	+12.38%	+12.38%	+9.25%	+8.23%
300000	*P@5*	0.2435	0.2475	0.2475	0.2297	0.2415	0.2376	0.2376
	Gaussian				+6.01%	+0.83%	+2.48%	+2.48%
	Rate		*Triangle*		+7.75%	+2.48%	+4.17%	+4.17%
				Cosine	+7.75%	+2.48%	+4.17%	+4.17%

(b) Precision after retrieving 10 documents *(P@10)*.

#docs	P	EXT/INT			EXT	INT		
		Gaussian	*Triangle*	*Cosine*		*Gaussian*	*Triangle*	*Cosine*
100000	*P@10*	0.1417	0.1406	0.1406	0.1296	0.1351	0.1351	0.1351
	Gaussian				**+9.34%**	+4.89%	+4.89%	+4.89%
	Rate		*Triangle*		+8.49%	+4.07%	+4.07%	+4.07%
				Cosine	+8.49%	+4.07%	+4.07%	+4.07%
200000	*P@10*	0.1979	0.1979	0.1979	0.1835	0.1824	0.1824	0.1835
	Gaussian				+7.85%	**+8.50%**	+7.41%	**+7.85%**
	Rate		*Triangle*		+7.85%	**+8.50%**	**+8.50%**	**+7.85%**
				Cosine	+7.85%	**+8.50%**	**+8.50%**	**+7.85%**
300000	*P@10*	0.2099	0.2089	0.2079	0.1970	0.2009	0.2059	0.2069
	Gaussian				+6.55%	+4.48%	+1.94%	+1.45%
	Rate		*Triangle*		+6.04%	+3.98%	+1.46%	+0.97%
				Cosine	+5.53%	+3.38%	+0.97%	+0.48%

to 0.1979 over EXT, INT (Gaussian), INT (Triangle) and INT (Cosine), respectively.

In terms of mean average precision, we notice that the proposed approach EXT/INT reports the best results in all the sub-collections (see Table 5.6), e.g. on the 300000 sub-collection, EXT/INT using Gaussian function outperforms EXT, INT(Gaussian), INT(Triangle) and INT(Cosine), around 3%, 4%, 5% and 5%, respectively.

In the second set of experiments, we evaluate and compare the results of the proposed approach (EXT/INT), which uses both the external and internal correlations, with those obtained by BIM and Rocchio; where we compute the precision values after retrieving 5 and 10 documents. In this experiment, the

TABLE 5.6: Comparing the effectiveness of EXT/INT, EXT and INT methods in terms of MAP.

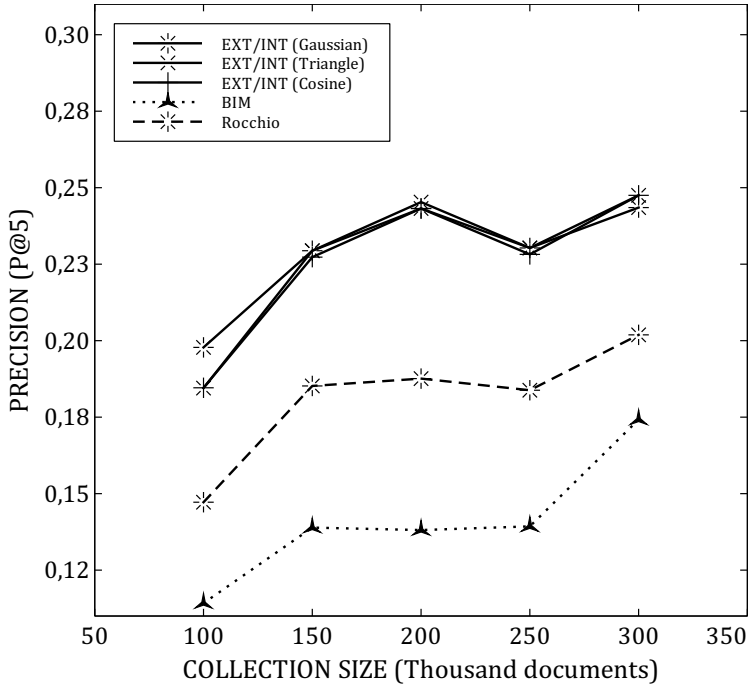
#docs	**MAP**	**EXT/INT**			**EXT**	**INT**		
		Gaussian	*Triangle*	*Cosine*		*Gaussian*	*Triangle*	*Cosine*
100000	*MAP*	0.1823	0.1781	0.1791	0.1658	0.1686	0.1719	0.1713
	Rate	*Gaussian*	*Triangle*	*Cosine*	**+9.95%**	+8.13%	+6.05%	+6.42%
					+7.42%	+5.63%	+3.61%	+3.97%
					+8.02%	+6.23%	+4.19%	+4.55%
200000	*MAP*	0.1663	0.1684	0.1686	0.1549	0.1531	0.1535	0.1537
	Rate	*Gaussian*	*Triangle*	*Cosine*	+7.36%	+8.62%	+8.34%	+8.20%
					+8.72%	+9.99%	+3.90%	+9.56%
					+8.84%	**+10.12%**	**+9.84%**	**+9.69%**
300000	*MAP*	0.1617	0.1607	0.1607	0.1556	0.1554	0.1526	0.1527
	Rate	*Gaussian*	*Triangle*	*Cosine*	+3.92%	+4.05%	+5.96%	+5.89%
					+3.28%	+3.41%	+5.31%	+5.24%
					+3.28%	+3.41%	+5.31%	+5.24%

parameters σ, R and the number of expansion keywords are set to 5, 10 and 10, respectively. Figure 5.1 shows the precision values for the EXT/INT, BIM and Rocchio techniques.

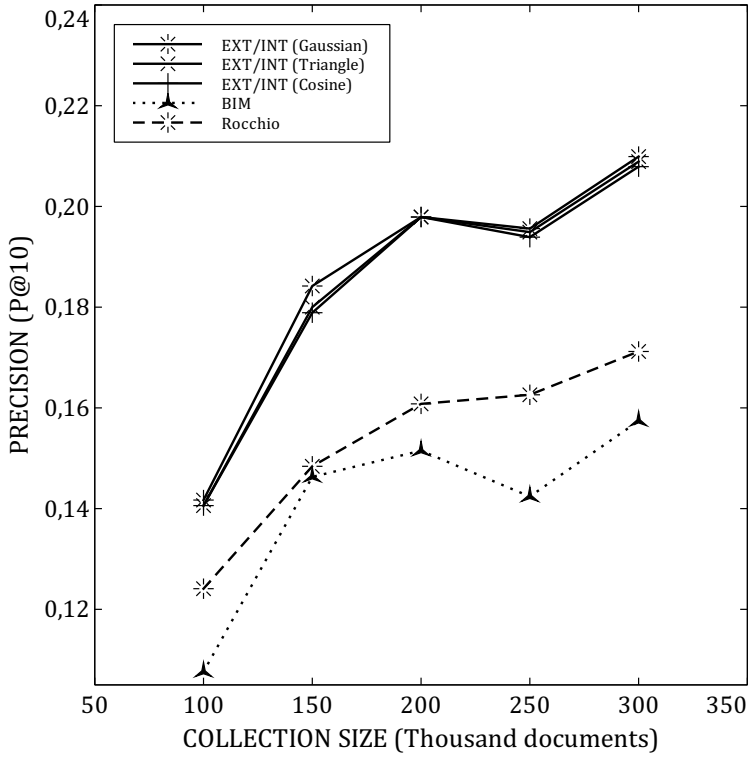
From Figure 5.1(a), we can see a clear superiority of the proposed approach EXT/INT compared to Rocchio, and this superiority is more significant in comparison with BIM technique. It is obviously seen from Figure 5.1(a) that the proposed approach manages to improve the search results, after retrieving 5 documents, in all the sub-collections; e.g. on the 300000 sub-collection, EXT/INT using Cosine shows great improvement of $+42.08\%$ over BIM and $+22.59\%$ over Rocchio. Despite the superiority shown in Figure 5.1(b), the results are not similar to those observed in Figure 5.1(a). Nevertheless, the precision values of the proposed approach after retrieving 10 documents are the best in all the sub-collections.

Through Table 5.5 and Figure 5.1, we can conclude that the proposed method EXT/INT succeeds to improve the ranking of the relevant documents and makes them in the first place. The precision values of the proposed system, after retrieving 5 documents, show a clear and significant superiority in front of each of EXT, INT, BIM and Rocchio techniques. This confirms the effectiveness of the EXT/INT approach.

In the next phase of testing, we compute the mean average precision to evaluate the retrieval effectiveness of the EXT/INT and the pseudo-relevance



(a) Precision after retrieving 5 documents (P@5).



(b) Precision after retrieving 10 documents (P@10).

FIGURE 5.1: Effectiveness comparison of the EXT/INT approach to the BIM and Rocchio methods in terms of precision.

feedback methods (see Table 5.7). The parameters σ, R and the number of expansion keywords are set to 5, 10 and 10, respectively. Furthermore, we use the two-tailed t-test to measure the statistical significance of differences between the MAP values.

TABLE 5.7: Mean Average Precision (MAP) results of EXT/INT, BIM and Rocchio methods.

#docs	**MAP**	**EXT/INT**			**BIM**	**Rocchio**
		Gaussian	*Triangle*	*Cosine*		
100000	*MAP*	0.1823	0.1781	0.1791	0.1253	0.1524
	Rate	*Gaussian*	*Triangle*	*Cosine*	**+45.49%**∗ +42.14%∗ +42.94%∗	+19.62%∗ +16.86% +17.52%
150000	*MAP*	0.1763	0.1784	0.1781	0.1285	0.1540
	Rate	*Gaussian*	*Triangle*	*Cosine*	+37.20%∗ +38.83%∗ +38.60%∗	+14.48% +15.84% +15.65%
200000	*MAP*	0.1663	0.1684	0.1686	0.1174	0.1346
	Rate	*Gaussian*	*Triangle*	*Cosine*	+41.65%∗ +43.44%∗ +43.61%∗	+23.55%∗ +25.11%∗ **+25.26%**∗
250000	*MAP*	0.1599	0.1585	0.1585	0.1204	0.1302
	Rate	*Gaussian*	*Triangle*	*Cosine*	+32.81%∗ +31.64%∗ +31.64%∗	+22.81%∗ +21.74%∗ +21.74%∗
300000	*MAP*	0.1617	0.1607	0.1607	0.1344	0.1333
	Rate	*Gaussian*	*Triangle*	*Cosine*	+20.31%∗ +19.57%∗ +19.57%∗	+21.31%∗ +20.56%∗ +20.56%∗

The $*$ indicates the difference is statistically significant, p-value < 0.05 with two-tailed t-test.

Table 5.7 shows a clear advantage of the EXT/INT approach compared to the BIM and Rocchio approaches. The improvements over BIM and Rocchio are statistically significant in 25 out of 30 cases ($p < 0.05$), and 30 out of 30 improvements are positive; e.g. on the 300000 sub-collection, EXT/INT (Gaussian) outperforms BIM by $+20.31\%$ and Rocchio by $+21.31\%$, while EXT/INT (Triangle) and EXT/INT (Cosine) outperform BIM by $+19.57\%$ and Rocchio by $+20.56\%$, respectively.

In the final set of experiments, we compare the performance of EXT/INT with BIM and Rocchio techniques while varying the values of σ, R and the number of expansion keywords. As a first step, we change the value of σ from 5 to 30 and maintain the parameter R and the number of expansion keywords constants at 10 (see Table 5.8). In the second step, we set σ and the number

of expansion keywords to 5 and 10, and vary the value of R to 20 and 50 (see Table 5.9). In the final step, we compare EXT/INT with BIM and Rocchio while varying the number of expansion terms from 10 to 5 and tune the parameters σ and R to 5 and 10, respectively (see Table 5.10).

TABLE 5.8: Comparing the performance of EXT/INT, BIM and Rocchio methods ($\sigma = 30$).

#docs	**P**	**EXT/INT**			**BIM**	**Rocchio**
		Gaussian	*Triangle*	*Cosine*		
100000	*P@5*	**0.1978**	0.1956	0.1956	0.1142	0.1472
	P@10	**0.1428**	**0.1428**	**0.1428**	0.1076	0.1241
	MAP	**0.1817**	0.1809	**0.1817**	0.1253∗	0.1524
200000	*P@5*	**0.2453**	0.2432	**0.2453**	0.1381	0.1876
	P@10	**0.1979**	**0.1979**	**0.1979**	0.1515	0.1608
	MAP	0.1654	**0.1662**	0.1658	0.1174∗	0.1346∗
300000	*P@5*	0.2415	0.2435	**0.2455**	0.1742	0.2019
	P@10	0.2069	**0.2079**	0.2069	0.1574	0.1712
	MAP	**0.1617**	0.1609	0.1614	0.1344∗	0.1333∗

TABLE 5.9: Precision and MAP results of EXT/INT, BIM and Rocchio approaches.

(a) Effectiveness comparison of EXT/INT with the baseline, $R = 20$.

#docs	**P/MAP**	**EXT/INT**			**BIM**	**Rocchio**
		Gaussian	*Triangle*	*Cosine*		
100000	*P@5*	**0.1978**	0.1846	0.1846	0.1230	0.1582
	MAP	**0.1813**	0.1785	0.1790	0.1358∗	0.1512
200000	*P@5*	**0.2453**	0.2412	0.2412	0.1546	0.2000
	MAP	0.1641	0.1662	**0.1680**	0.1405	0.1339∗
300000	*P@5*	0.2415	**0.2455**	**0.2455**	0.1742	0.2198
	MAP	0.1633	**0.1655**	0.1645	0.1496	0.1291∗

(b) Effectiveness comparison of EXT/INT with the baseline, $R = 50$.

#docs	**P/MAP**	**EXT/INT**			**BIM**	**Rocchio**
		Gaussian	*Triangle*	*Cosine*		
100000	*P@5*	**0.1890**	0.1824	0.1824	0.1406	0.1274
	MAP	**0.1786**	0.1742	0.1747	0.1427∗	0.1394∗
200000	*P@5*	**0.2329**	0.2309	0.2288	0.1711	0.1649
	MAP	0.1597	**0.1633**	0.1628	0.1383∗	0.1169∗
300000	*P@5*	0.2396	**0.2435**	**0.2435**	0.1762	0.1920
	MAP	0.1608	0.1618	**0.1619**	0.1447	0.1121∗

Once again, the proposed approach EXT/INT achieves better performance than the others, even though the number of pseudo-relevant documents R, the parameter σ and the number of expansion terms are varied. In terms of precision, the EXT/INT results are the best in all the sub-collections and in terms

TABLE 5.10: Comparing the performance of EXT/INT, BIM and Rocchio methods.

#docs	P/MAP	EXT/INT			BIM	Rocchio
		Gaussian	*Triangle*	*Cosine*		
100000	*P@5*	**0.1736**	0.1692	0.1670	0.0857	0.1670
	P@10	**0.1384**	0.1362	0.1362	0.1087	0.1252
	MAP	0.1717	0.1726	**0.1727**	0.1150∗	0.1626
200000	*P@5*	0.2185	**0.2247**	**0.2247**	0.1195	0.2000
	P@10	**0.1845**	0.1835	0.1835	0.1525	0.1639
	MAP	**0.1543**	0.1529	0.1517	0.0993∗	0.1465
300000	*P@5*	**0.2435**	0.2376	0.2356	0.1306	0.2099
	P@10	0.2029	**0.2079**	**0.2079**	0.1623	0.1772
	MAP	0.1540	**0.1559**	**0.1559**	0.1036∗	0.1338∗

of MAP, EXT/INT performs statistically significantly better than BIM and Rocchio in a considerable number of cases.

5.4.2 Swarm intelligence for query expansion

Contrary to the above testing, the whole collection is used in this section. In addition, all non-informative words such as prepositions, conjunctions, pronouns and very common verbs are disregarded during the indexing phase. Moreover, the most common morphological and inflectional suffixes are removed using a standard stemming algorithm. In addition, the weights of the words are calculated using the well-known Okapi BM25 term weighting function.

a. Parameter settings

Before proceeding to discuss the performance of the proposed approaches and compare them with BIM and Rocchio methods, we first fix the values of BA, FA and APSO parameters.

BA parameter settings:
As mentioned above, we tune the values of the following BA parameters: f_{max}, r, A, $|\grave{Q}|$, N and T, which respectively represent the maximum frequency, the pulse emission rate, the loudness, the solution length, the population size, and the maximum number of generations. In this preliminary experiment, each of

these parameters is varied individually while keeping the remaining parameters constant. Regarding the parameters f_{max}, r and A, their values are varied following the guidelines proposed in several studies of the literature [84]. During this test, the number of pseudo-relevant documents $|R|$ is tuned at 10, 30 and 50. We use both the precision values after retrieving 5 documents ($P@5$) and the mean average precision metric to determine the best values for each parameter. Figures 5.2(a), 5.2(b), 5.2(c), 5.2(d), 5.2(e) and 5.2(f) present the precision values obtained when setting the BA parameters.

In Figure 5.2(a), we set r, A, $|\grave{Q}|$, N and T to 0.5, 0.5, 5, 5 and 10, and systematically vary the frequency f_{max} from 0.5 to 3.5 in increments of 0.5. As illustrated in Figure 5.2(a), the appropriate values of f_{max} which give the highest performance are 1.5 and 2.5. In Figure 5.2(b), we fix f_{max}, A, $|\grave{Q}|$, N and T to 2.5, 0.5, 5, 5 and 10, and adjust the parameter r from 0.1 to 0.9. According to Figure 5.2(b), the best value of r is 0.7. In Figure 5.2(c), the parameters f_{max}, r, $|\grave{Q}|$, N and T are respectively tuned to 2.5, 0.7, 5, 5 and 10, while A is altered from 0.1 to 0.9 in increments of 0.2. From Figure 5.2(c), it can be seen that the suitable values of A are 0.1 and 0.7. In Figure 5.2(d), the solution length $|\grave{Q}|$ is varied from 2 to 14, while the parameters f_{max}, r, A, N and T are set to 2.5, 0.7, 0.7, 5 and 10. As shown in Figure 5.2(d), the satisfactory values of $|\grave{Q}|$ are 6 and 8. In Figure 5.2(e), we tune f_{max}, r, A, $|\grave{Q}|$ and T to 2.5, 0.7, 0.7, 6 and 10, and gradually vary N from 5 to 60. According to Figure 5.2(e), the best values of N which bring the best retrieval effectiveness are 20 and 40. Finally, in Figure 5.2(f), the parameters f_{max}, r, A, $|\grave{Q}|$ and N are fixed to 2.5, 0.7, 0.7, 6 and 20, and the maximum number of generations T is adjusted from 10 to 120. As it can be seen from Figure 5.2(f), the appropriate values of T are 20 and 40. The MAPs results are shown in Figure 5.3. Similar to the above results, the best mean average precision results are achieved when $f_{max} = 2.0$ or 2.5, $r = 0.5$ or 0.7, $A = 0.3$ or 0.7, $|\grave{Q}| = 2$ or 6, $N = 10$ or 20 and $T = 40$ or 120.

Based on these preliminary experiments, the values of the BA parameters used in all the following tests are presented in Table 5.11.

FA parameter settings:

We tune the values of the following FA parameters: γ, α_0, θ, $|\grave{Q}|$, N and T, which

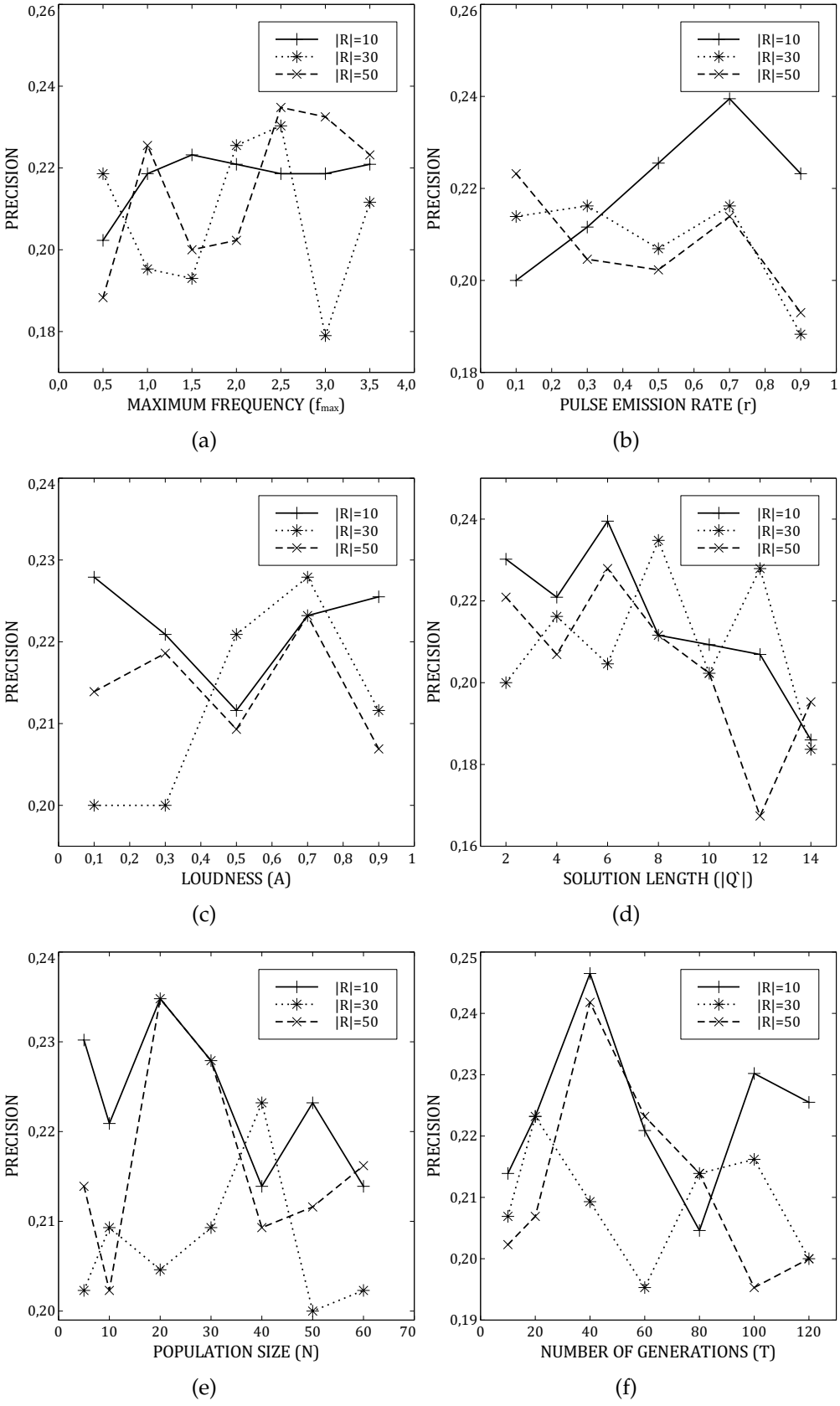


FIGURE 5.2: The precision values achieved when fixing the BA parameters

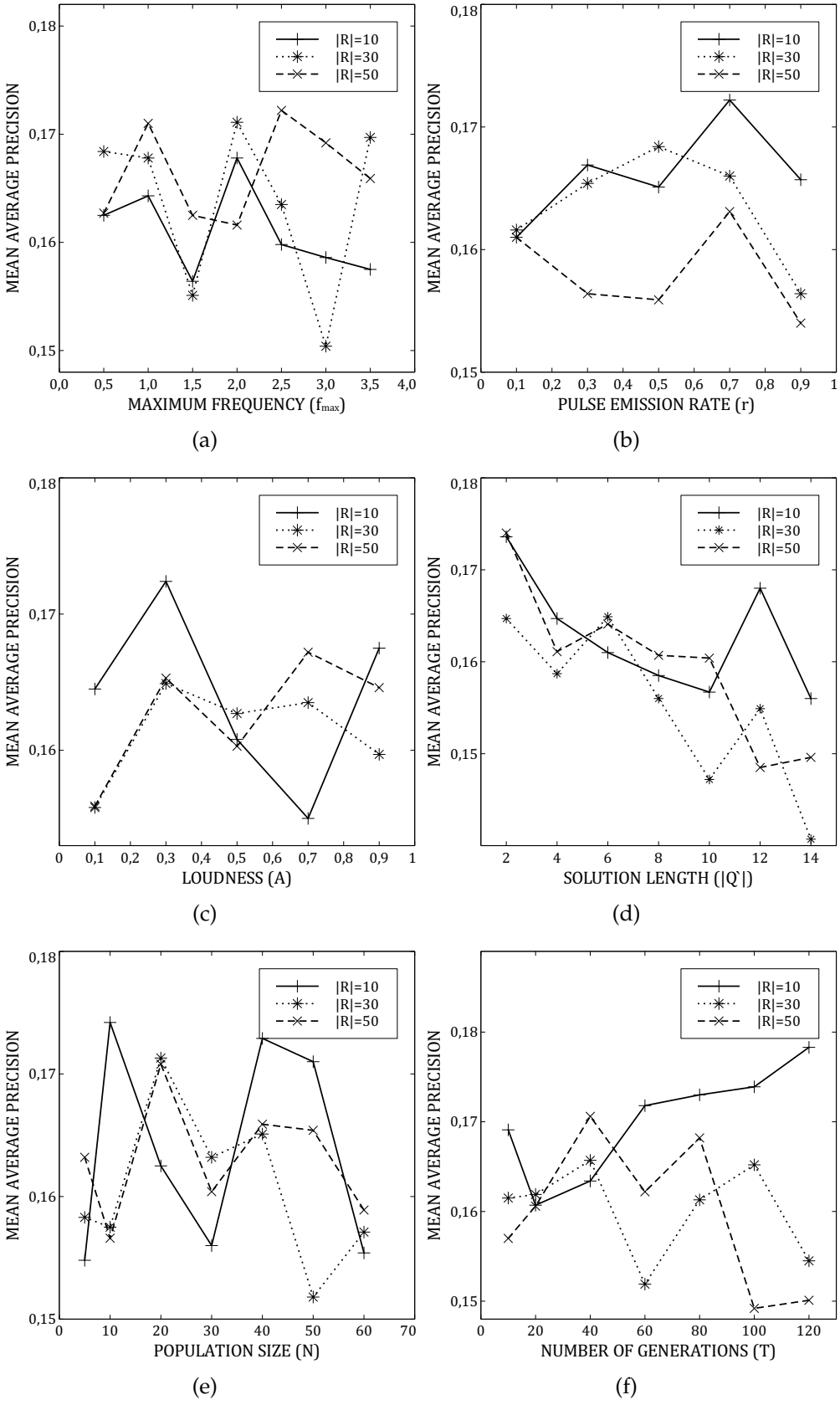


FIGURE 5.3: The MAP values obtained when fixing the BA parameters

TABLE 5.11: BA parameter settings.

Maximum frequency (f_{max})	2.5
Pulse emission rate (r)	0.7
Loudness (A)	0.7
Solution length ($\|\grave{Q}\|$)	6
Population size (N)	20
Number of generations (T)	40

respectively represent the light absorption coefficient, the initial randomization, the randomness reduction constant, the solution length, the population size, and the maximum number of generations. In this preliminary experiment, each of these parameters is varied individually while keeping the remaining parameters constant. Regarding the parameters γ, α_0 and θ, their values are varied following the guidelines proposed in several studies of the literature [84]. During this test, the number of pseudo-relevant documents $|R|$ is tuned at 10, 30 and 50. We use both the precision values after retrieving 5 documents ($P@5$) and the mean average precision metric to determine the best values for each parameter. Figures 5.4(a), 5.4(b), 5.4(c), 5.4(d), 5.4(e) and 5.4(f) present the precision values obtained when setting the FA parameters.

In Figure 5.4(a), we set α_0, θ, $|\grave{Q}|$, N and T to 1.0, 0.95, 5, 5 and 10, and systematically vary the frequency γ from 0.4 to 1.6 in increments of 0.2. As illustrated in Figure 5.4(a), the appropriate values of γ which give the highest performance are between 1.0 and 1.6. In Figure 5.4(b), we fix γ, θ, $|\grave{Q}|$, N and T to 1.0, 0.95, 5, 5 and 10, and adjust the parameter α_0 from 0.1 to 1.0. According to Figure 5.4(b), the best value of α_0 are 0.6 and 0.8. In Figure 5.4(c), the parameters γ, α_0, $|\grave{Q}|$, N and T are respectively tuned to 1.0, 0.6, 5, 5 and 10, while θ is altered from 0.87 to 0.99 in increments of 0.02. From Figure 5.4(c), it can be seen that the suitable values of θ are 0.91 and 0.95. In Figure 5.4(d), the solution length $|\grave{Q}|$ is varied from 2 to 14, while the parameters γ, α_0, θ, N and T are set to 1.0, 0.6, 0.91, 5 and 10. As shown in Figure 5.4(d), the satisfactory values of $|\grave{Q}|$ are 4 and 6. In Figure 5.4(e), we tune γ, α_0, θ, $|\grave{Q}|$ and T to 1.0, 0.6, 0.91, 4 and 10, and gradually vary N from 5 to 60. According to Figure 5.4(e), the best values of N which bring the best retrieval effectiveness are 30 and 40. Finally, in Figure 5.4(f), the parameters γ, α_0, θ, $|\grave{Q}|$ and N are fixed to 1.0, 0.6, 0.91, 4

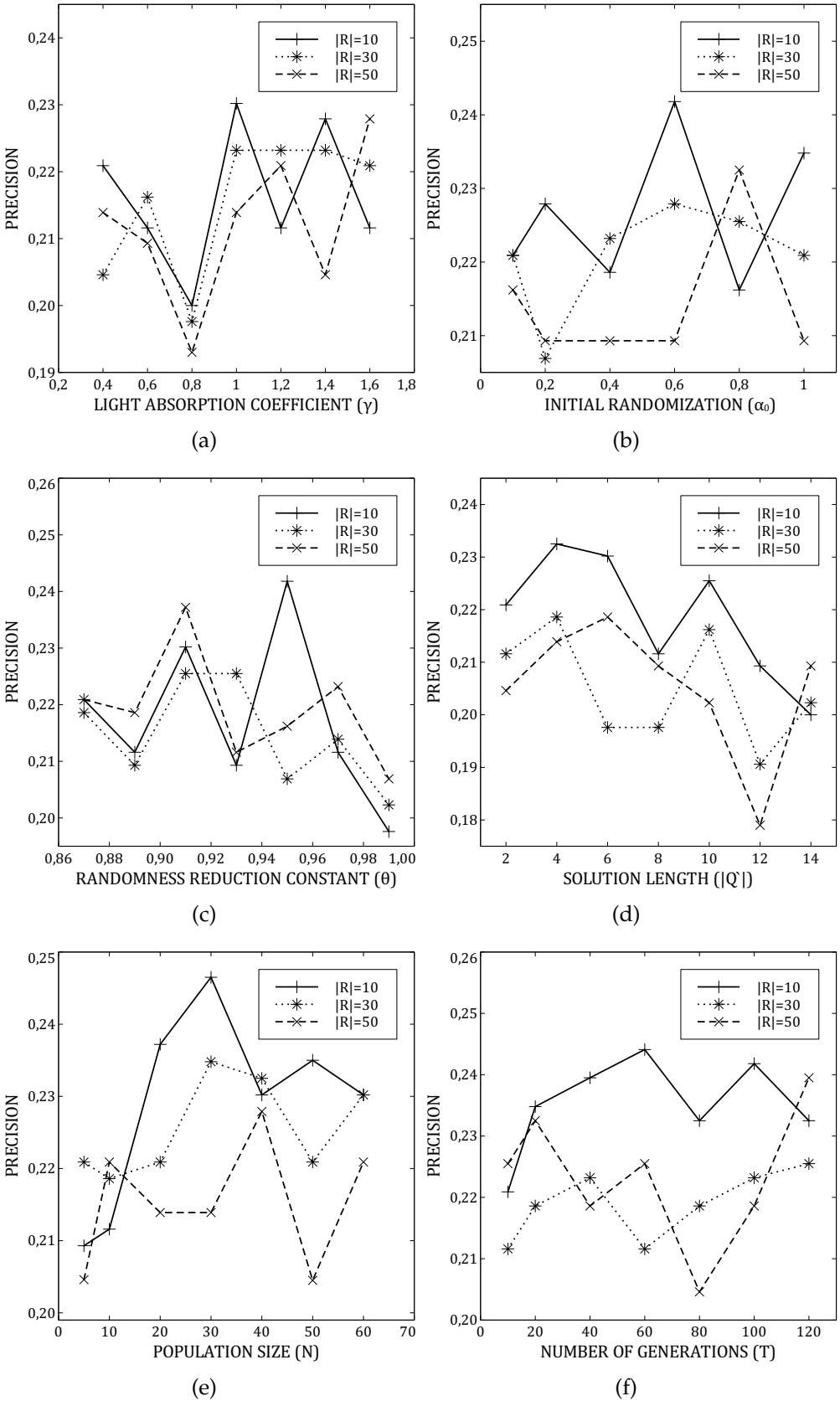


FIGURE 5.4: The precision values achieved when fixing the FA parameters

and 30, and the maximum number of generations T is adjusted from 10 to 120. As it can be seen from Figure 5.4(f), the appropriate values of T are 60 and 120. The MAPs results are shown in Figure 5.5. Similar to the above results, the best mean average precision results are achieved when $\gamma = 1.0$ or 1.2, $\alpha_0 = 0.6$ or 1.0, $\theta = 0.91$ or 0.93, $|\grave{Q}| = 4$, $N = 30$ or 60 and $T = 20$ or 60.

Based on these preliminary experiments, the values of the FA parameters used in all the following tests are presented in Table 5.12.

TABLE 5.12: FA parameter settings.

Light absorption coefficient (γ)	1.0
Initial randomization (α_0)	0.6
Randomness reduction constant (θ)	0.91
Solution length ($\|\grave{Q}\|$)	4
Population size (N)	30
Number of generations (T)	60

APSO parameter settings:

In this section, we tune the values of the following APSO parameters: α_0, γ, $|\grave{Q}|$, N, and T, which respectively represent the initial randomization parameter, the control parameter, the solution length, the population size, and the maximum number of generations. In this preliminary experiment, each of these parameters is varied individually while keeping the remaining parameters constant. Regarding the parameters α_0 and γ, their values are varied following the guidelines proposed in several studies of the literature [84]. During this test, the number of pseudo-relevant documents $|R|$ is tuned at 10, 30 and 50. We use both the precision values after retrieving 5 documents ($P@5$) and the mean average precision metric to determine the best values for each parameter. Figures 5.6(a), 5.6(b), 5.6(c), 5.6(d) and 5.6(e) present the precision values obtained when setting the APSO parameters.

In Figure 5.6(a), we set γ, $|\grave{Q}|$, N and T to 0.95, 5, 5 and 10, and systematically vary the initial randomization parameter α_0 from 0.1 to 1.0. As illustrated in Figure 5.6(a), the appropriate values of α_0 which give the highest performance are 0.2 and 1.0. In Figure 5.6(b), we fix α_0, $|\grave{Q}|$, N and T to 1.0, 5, 5 and 10, and adjust the parameter γ from 0.87 to 0.99 in increments of 0.02, According to Figure 5.6(b), the best values of γ are 0.89, 0.91 and 0.99. In Figure 5.6(c), α_0, γ,

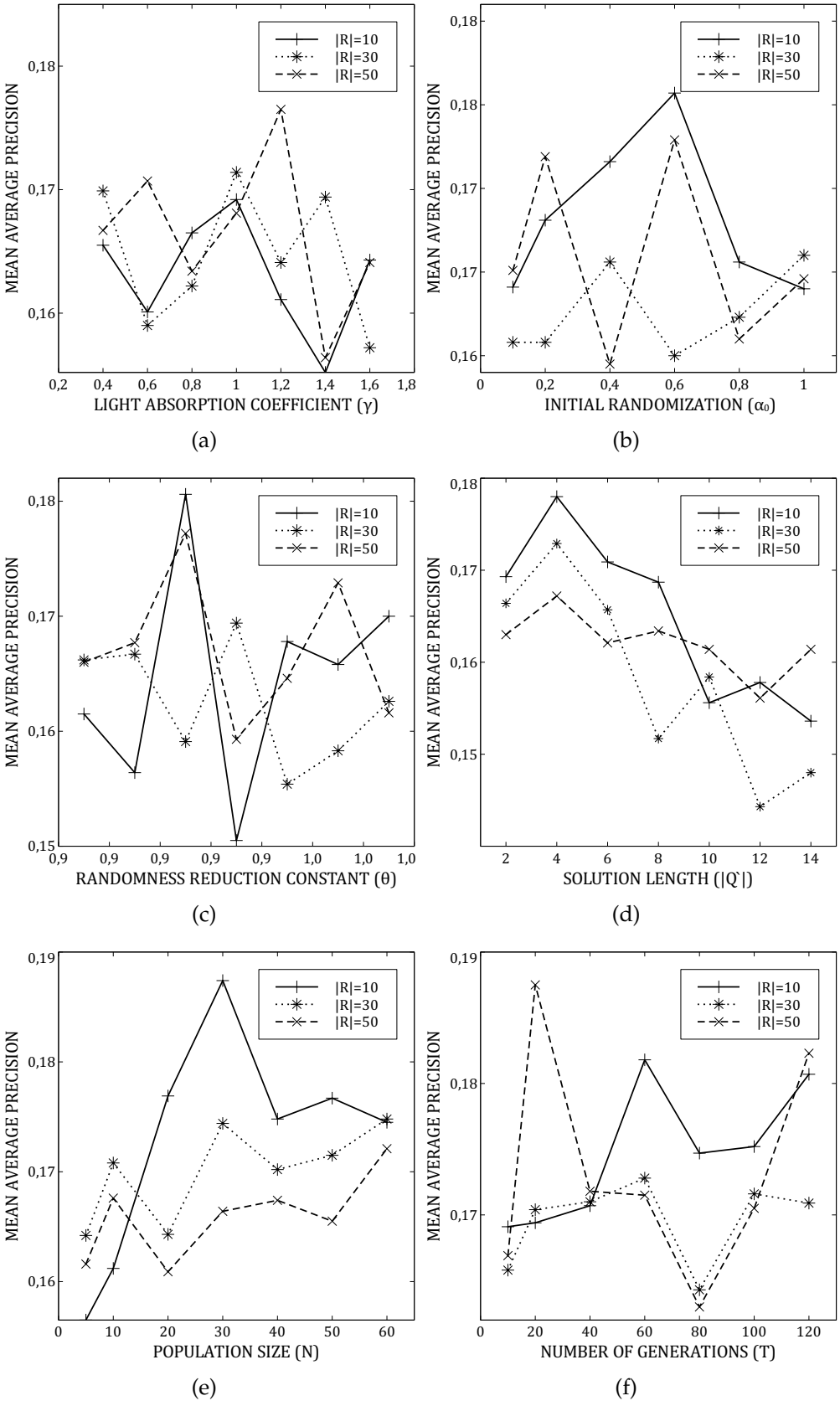


FIGURE 5.5: The MAP values obtained when fixing the FA parameters

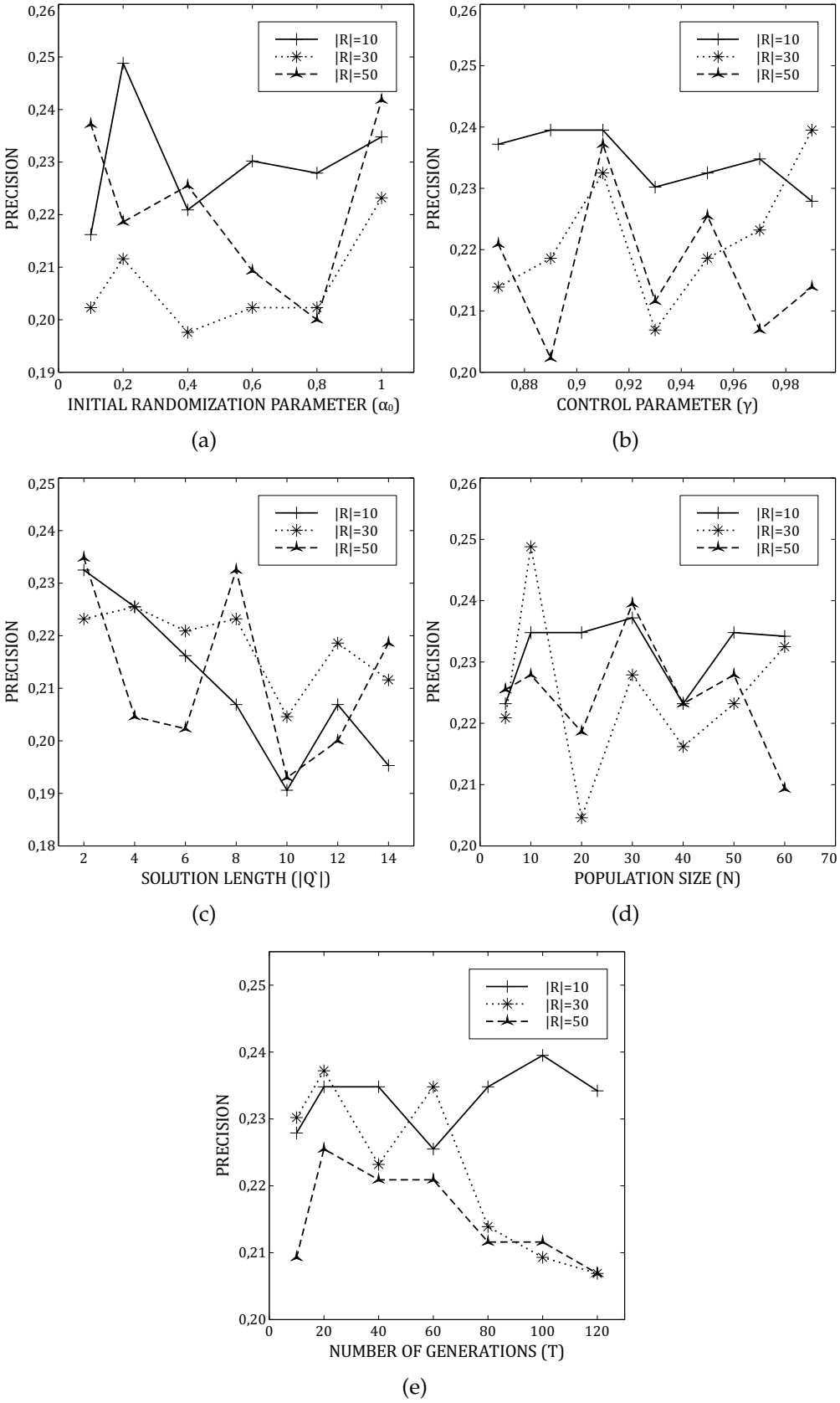


FIGURE 5.6: The precision values achieved when fixing the APSO parameters

N, and T are respectively tuned to 1.0, 0.91, 5 and 10, while the solution length $|\grave{Q}|$ is altered from 2 to 14 in increments of 2. From Figure 5.6(c), it can be seen that the suitable values of $|\grave{Q}|$ are 2 and 4. In Figure 5.6(d), the value of N is varied from 5 to 60 and the parameters α_0, γ, $|\grave{Q}|$, and T are set to 1.0, 0.91, 4 and 10. As shown in Figure 5.6(d), the satisfactory values of N are 10 and 30. Finally, in Figure 5.6(e), we tune α_0, γ, $|\grave{Q}|$, and N to 1.0, 0.95, 4 and 30, and gradually vary T from 10 to 120. According to Figure 5.6(e), the best values of T which bring the best retrieval effectiveness are 20 and 100. The MAPs results are shown in Figure 5.7. Similar to the above results, the best mean average precision results are achieved when $\alpha_0 = 0.2$ or 1.0, $\gamma = 0.89$ or 0.91, $|\grave{Q}| = 2$ or 4, $N = 10$ or 30 and $T = 20$ or 60.

Based on these preliminary experiments, the values of the APSO parameters, used in all the following tests, are presented in Table 5.13.

TABLE 5.13: APSO parameter settings

Initial randomization parameter (α_0)	1.0
Control parameter (γ)	0.91
Solution length ($|\grave{Q}|$)	4
Population size (N)	30
Number of generations (T)	20

b. Testing results

In this section, we compare the results obtained by the proposed approaches, which used BA, FA and APSO; with those achieved by BIM and Rocchio methods.

BA results:

In the first set of experiments, the number of pseudo-relevant documents is varied from 10 to 300 in increments of 50. Figure 5.8 shows the precision values of each method after retrieving 5 ($P@5$) and 10 ($P@10$) documents.

Through Figure 5.8(a), we can clearly see that the proposed approach produces the highest $P@5$ values and achieves substantial improvement over BIM and Rocchio methods, e.g. for $|R| = 50$, precision at 5 retrieved documents

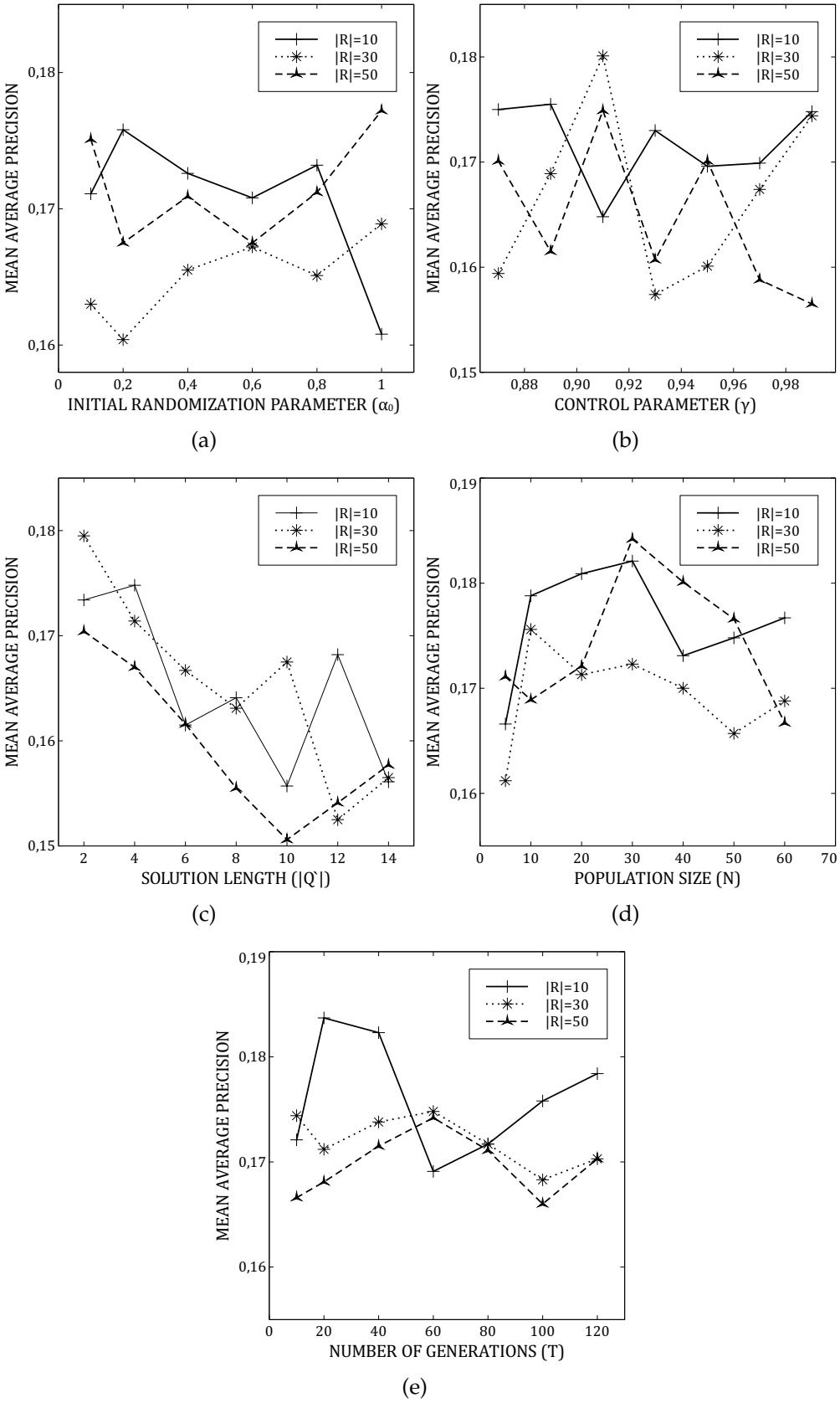
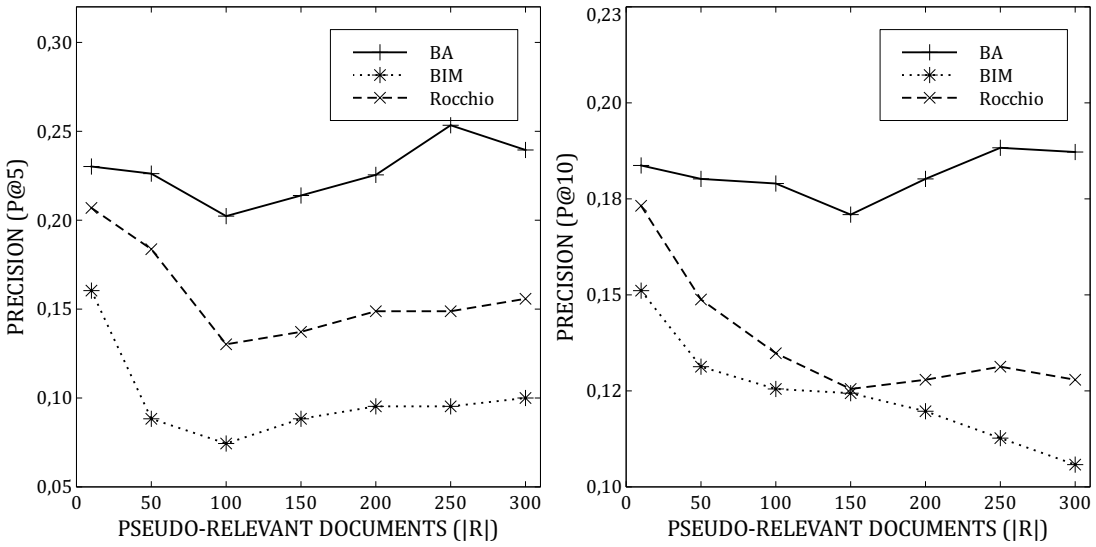


FIGURE 5.7: The MAP values obtained when fixing the APSO parameters



(a) (b)

FIGURE 5.8: Comparing the effectiveness of BA, BIM and Rocchio in terms of precision

improves from 0.0883 and 0.1837 to 0.2262 over BIM ($+156.16\%$) and Rocchio ($+23.14\%$), respectively. Furthermore, for $|R| = 250$, there is an enhancement of 165.90% over BIM and 70.30% over Rocchio. Similarly, for $|R| = 50$, precision at 10 retrieved documents improves from 0.1313 and 0.1488 to 0.1802 over BIM ($+37.24\%$) and Rocchio ($+21.10\%$), respectively. In terms of mean average precision, we observe that the BA reports the best results against BIM and Rocchio methods (see Figure 5.9), e.g. for $|R| = 50$, the proposed approach outperforms BIM and Rocchio around 57.13% and 25.97%, respectively.

In the next phase of testing, the number of words in $|\grave{Q}|$ is varied from 2 to 14 in increments of 2, while the number of pseudo-relevant documents $|R|$ is tuned at 10. Figure 5.10 presents the precision values achieved by the proposed approach, BIM and Rocchio after retrieving 5 and 10 documents.

From Figure 5.10, we can see a clear superiority of the proposed approach over Rocchio, and this superiority is more significant in comparison with BIM. It is obviously seen from Figure 5.10(a) that the proposed approach operates to improve the search results after retrieving 5 documents, e.g. for $|\grave{Q}| = 10$, the proposed approach shows improvement of 33.63% over BIM and 10.58% over Rocchio. It can be also observed from Figure 5.11 that the proposed approach achieves the highest values in terms of mean average precision, e.g. for

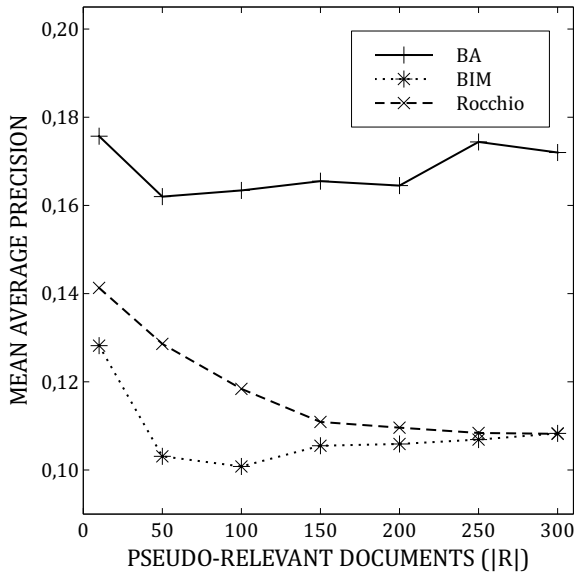


FIGURE 5.9: Mean average precision results of the BA, the BIM and Rocchio methods

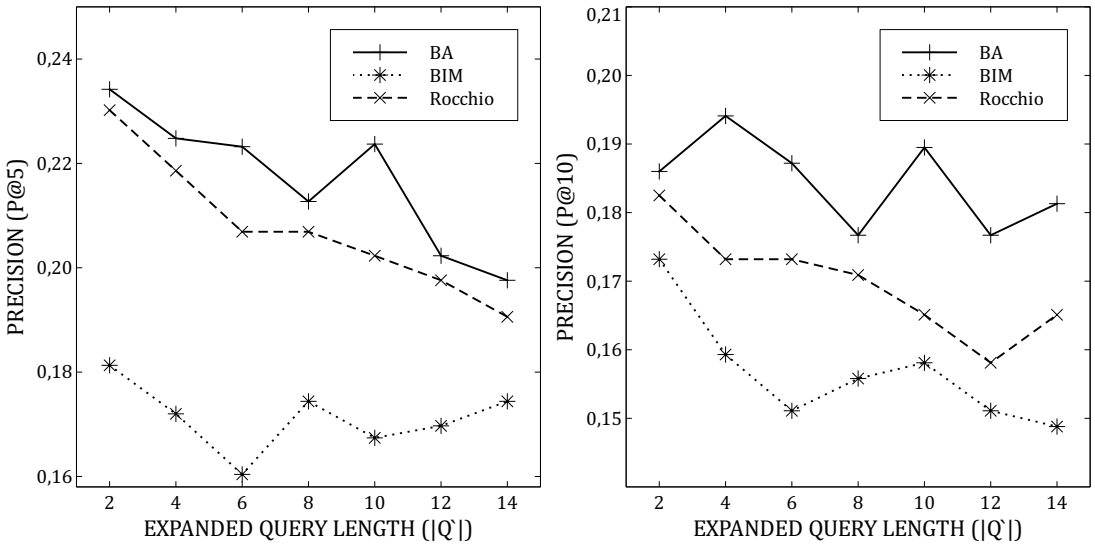


FIGURE 5.10: Effectiveness comparison of the proposed approach to the BIM and Rocchio methods in terms of precision

$|\grave{Q}| = 12$, there is an improvement of 26.92% over BIM and 20.18% over Rocchio.

FA results:

In the next set of experiments, we evaluate and compare the results obtained by the FA with those obtained by BIM and Rocchio methods. In these tests, the number of pseudo-relevant documents is varied from 10 to 300 in increments of 50 while the length of $\grave{Q}$ is set to 4, . Figure 5.12 shows the precision values

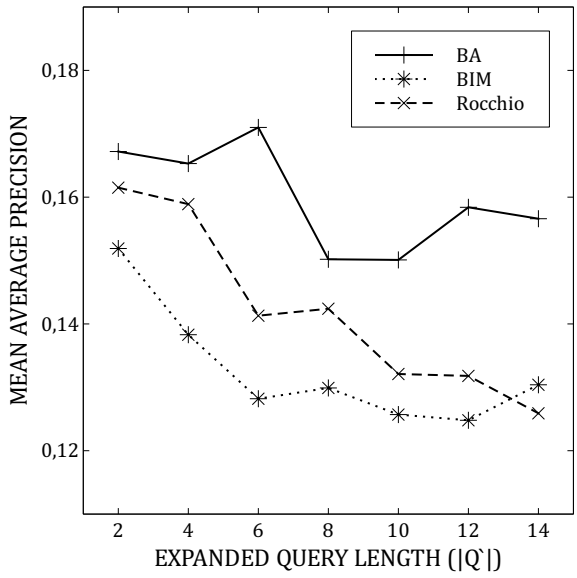


FIGURE 5.11: MAP results of BA, BIM and Rocchio

of each method after retrieving 5 and 10 documents.

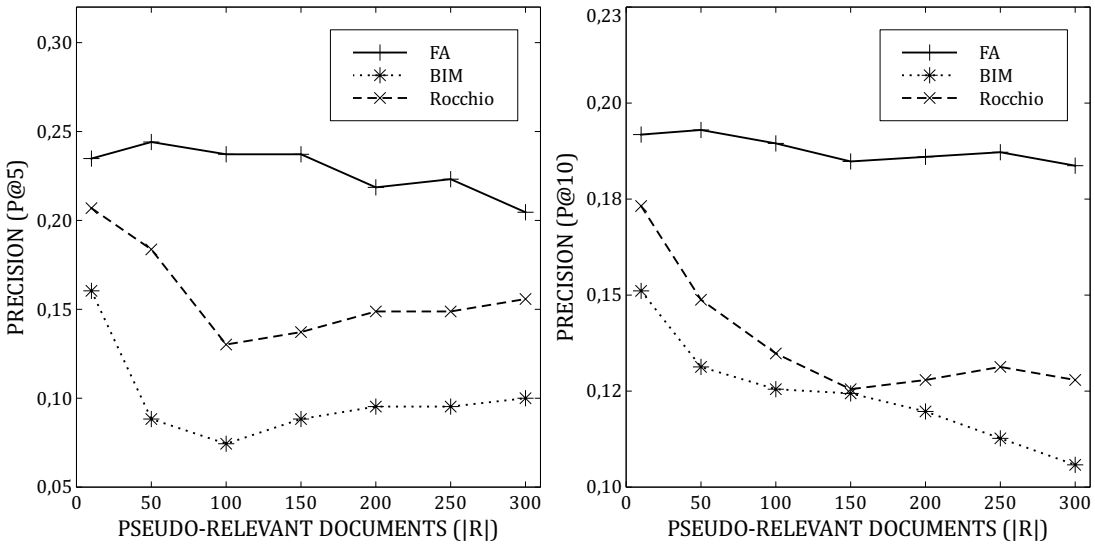


FIGURE 5.12: Comparing the effectiveness of FA, BIM and Rocchio in terms of precision

Through Figure 5.12(a), we can see obviously that the proposed approach produces the highest $P@5$ values and achieves significant improvement over BIM and Rocchio methods, e.g. for $|R| = 50$, there is an improvement of $+70.20\%$ over BIM and $+3.74\%$ over Rocchio. Similarly, precision at 10 retrieved documents improves from 0.1558 $(+19.38\%)$ and 0.1616 $(+15.09\%)$ to 0.1860 over BIM and Rocchio, respectively. In terms of mean average precision,

we observe that the FA reports the best results against BIM and Rocchio methods (Figure 5.13), e.g. for $|R| = 10$, the proposed approach outperforms BIM and Rocchio around $+29.42\%$ and $+12.64\%$, respectively.

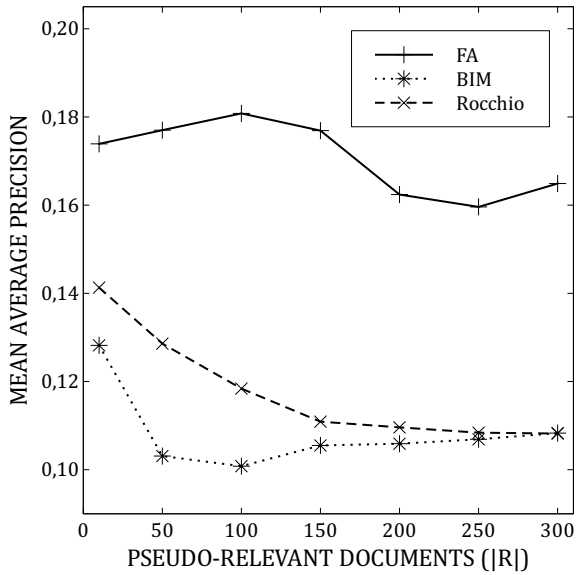


FIGURE 5.13: Mean average precision results of the FA, the BIM and Rocchio methods

In the next phase of testing, the number of words in $|\grave{Q}|$ is varied from 2 to 14 in increments of 2, while the number of pseudo-relevant documents $|R|$ is tuned at 10. Figure 5.14 presents the precision values achieved by the proposed approach, BIM and Rocchio after retrieving 5 and 10 documents.

From Figure 5.14(a), we can observe that the proposed approach manages to enhance the search results against Rocchio and this enhancement is more significant against BIM, e.g. for $|\grave{Q}| = 6$, the proposed approach shows improvement of $+47.88\%$ over BIM and $+14.64\%$ over Rocchio after retrieving 5 documents. It can be also observed from Figure 5.15 that the proposed approach achieves the highest values in terms of mean average precision, e.g. for $|\grave{Q}| = 6$, there is an improvement of $+41.02\%$ over BIM and $+27.95\%$ over Rocchio.

APSO results:

In this set of experiments, we compare the results obtained by the proposed APSO with those achieved by BIM and Rocchio methods. In these tests, the number of pseudo-relevant documents is varied from 10 to 300 in increments

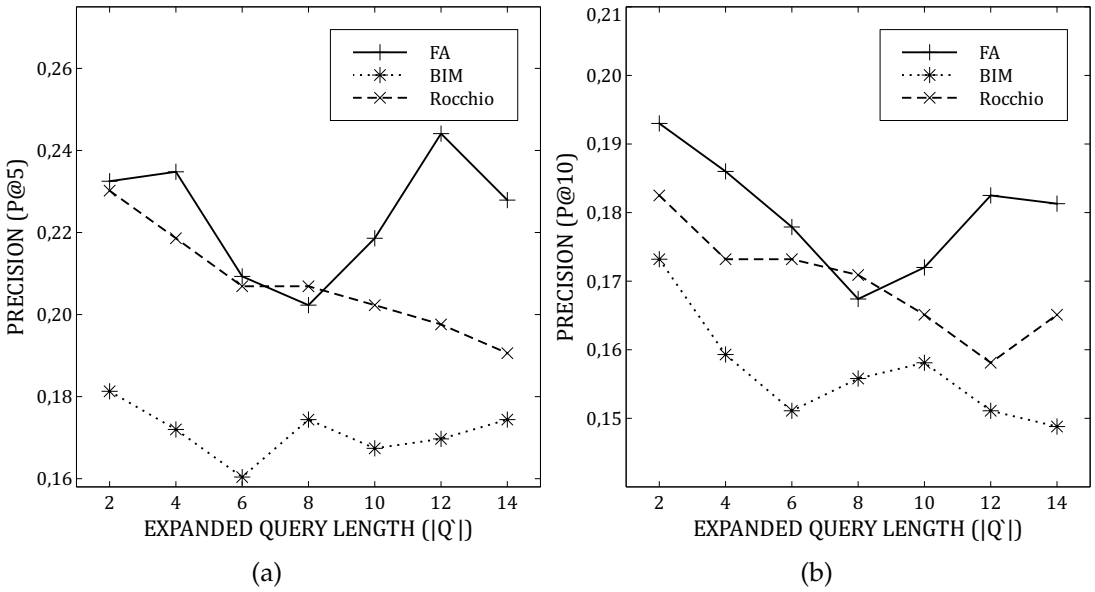


FIGURE 5.14: Effectiveness comparison of the proposed approach to the BIM and Rocchio methods in terms of precision

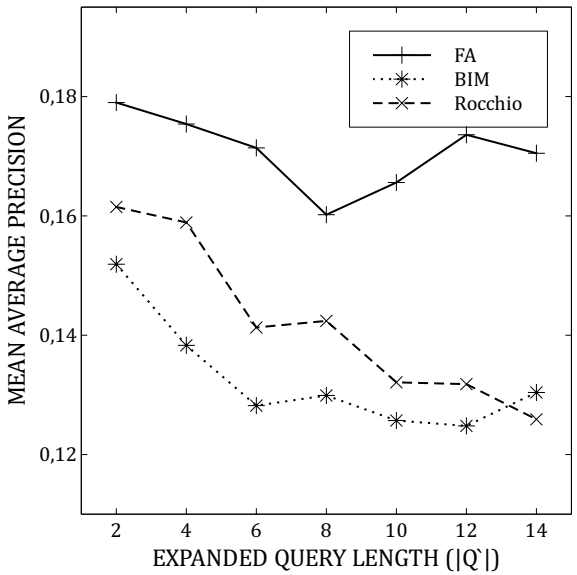


FIGURE 5.15: MAP results of FA, BIM and Rocchio

of 50. Figure 5.16 shows the precision values of each method after retrieving 5 ($P@5$) and 10 ($P@10$) documents.

Through Figure 5.16(a), we can clearly see that the proposed approach produces the highest $P@5$ values and achieves substantial improvement over BIM and Rocchio methods, e.g. for $|R| = 10$, precision at 5 retrieved documents improves from 0.1720 and 0.2186 to 0.2395 over BIM ($+39.24\%$) and Rocchio ($+9.56\%$), respectively. Furthermore, for $|R| = 200$, there is an enhancement of

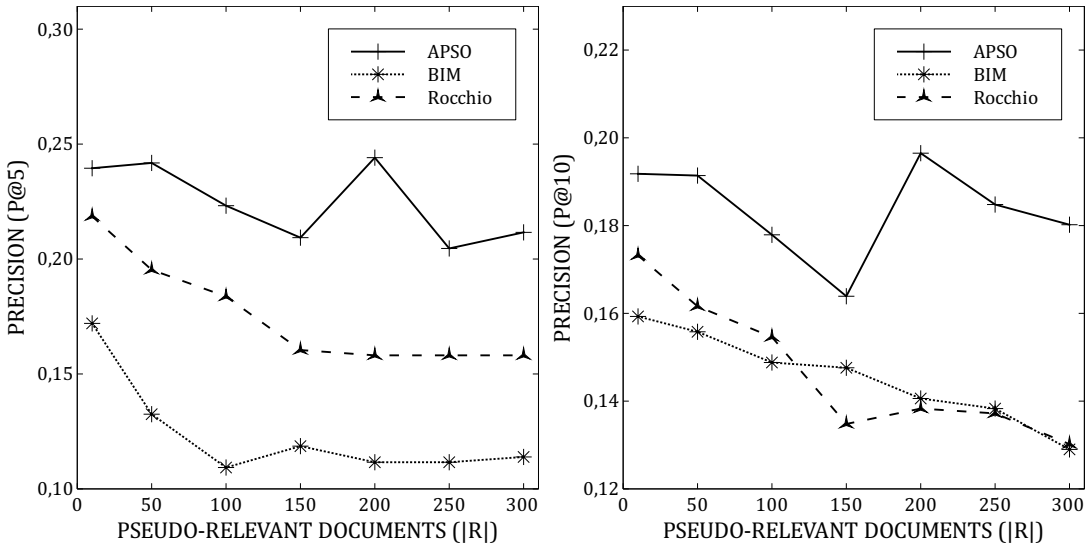


FIGURE 5.16: Comparing the effectiveness of APSO, BIM and Rocchio in terms of precision

118.72% over BIM and 54.39% over Rocchio. Similarly, for $|R| = 200$, precision at 10 retrieved documents improves from 0.1406 and 0.1383 to 0.1965 over BIM ($+39.75\%$) and Rocchio ($+42.08\%$), respectively.

In terms of mean average precision, we observe that the APSO reports the best results against BIM and Rocchio methods (see Figure 5.17), e.g. for $|R| = 50$, the proposed approach outperforms BIM and Rocchio around 15.82% and 5.97%, respectively.

In the next phase of testing, the number of words in $|\grave{Q}|$ is varied from 2 to 14 in increments of 2, while the number of pseudo-relevant documents $|R|$ is tuned at 10. Figure 5.18 presents the precision values achieved by the proposed approach, BIM and Rocchio after retrieving 5 and 10 documents.

From Figure 5.18, we can see a clear superiority of the proposed approach over Rocchio, and this superiority is more significant in comparison with BIM. It is obviously seen from Figure 5.18(a) that the proposed approach operates to improve the search results after retrieving 5 documents, e.g. for $|\grave{Q}| = 2$, the proposed approach shows improvement of 35.96% over BIM and 7.80% over Rocchio. It can be also observed from Figure 5.19 that the proposed approach achieves the highest values in terms of mean average precision, e.g. for $|\grave{Q}| = 8$, there is an improvement of 38.79% over BIM and 26.61% over Rocchio.

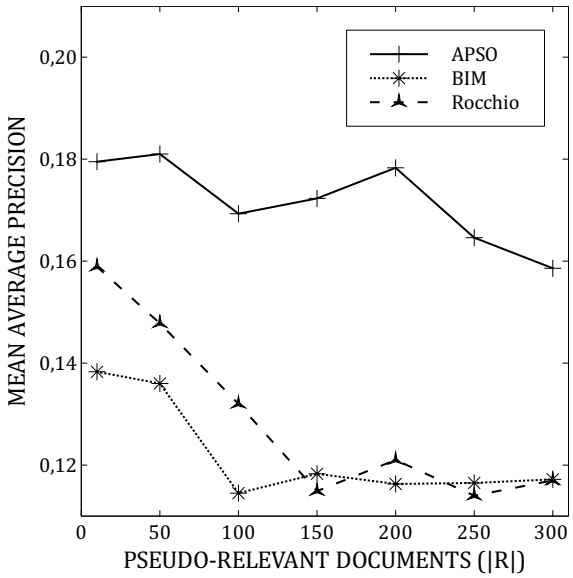


FIGURE 5.17: Mean average precision results of the APSO, the BIM and Rocchio methods

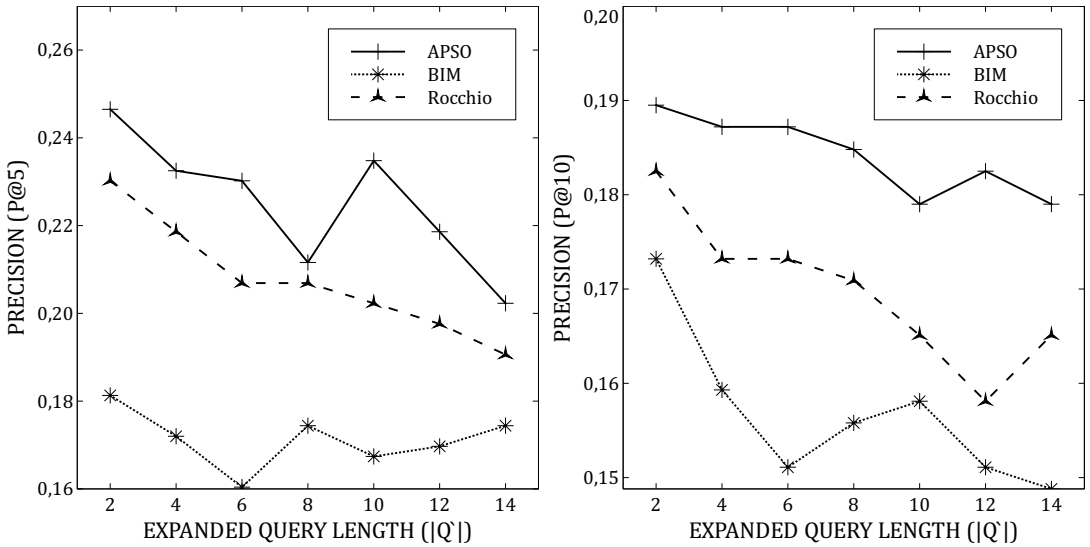


FIGURE 5.18: Effectiveness comparison of the proposed approach to the BIM and Rocchio methods in terms of precision

Efficiency comparison:

The computational complexity analysis of the proposed approaches in comparison to RSJ and Rocchio is shown in Table 5.14. Following the above experiments, in this comparison, the size of R is first varied from 10 to 300 (see Table 5.14(a)) and then the length of $\grave{Q}$ is varied from 2 to 14 (see Table 5.14(b)).

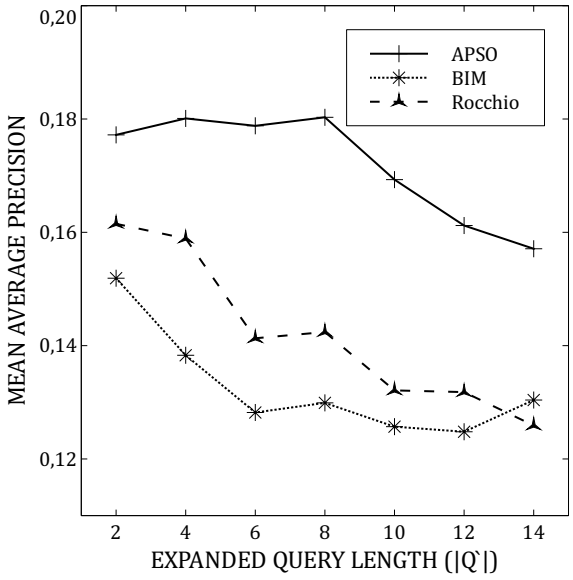


FIGURE 5.19: MAP results of APSO, BIM and Rocchio

TABLE 5.14: Efficiency comparison in terms of CPU time.

(a) CPU time comparison for different values of $|R|$ (seconds).

$\|\mathbf{R}\|$	**10**	**50**	**100**	**150**	**200**	**250**	**300**
SI	0.378	0.302	0.387	0.395	0.312	0.288	0.321
RSJ, Rocchio	11.189	13.614	12.599	15.327	16.462	18.862	19.735

(b) CPU time comparison for different values of $|\grave{Q}|$ (seconds).

$\|\grave{\mathbf{Q}}\|$	**2**	**4**	**6**	**8**	**10**	**12**	**14**
SI	0.214	0.251	0.386	0.447	0.493	0.533	0.579
RSJ, Rocchio	10.686	9.953	11.524	10.712	10.055	10.390	10.735

From Table 5.14, we can observe that the swarm intelligence algorithms achieves better results in all cases and gives great enhancement over RSJ and Rocchio in terms of CPU time.

Chapter 6

Conclusion

Query Expansion (QE) is one of the most powerful and effective method to improve the performance of information retrieval systems. This method aims to augment the original search queries with additional keywords that best characterize the users' needs.

Due to the dramatic increase of the volume of data available on the Internet, a huge number of QE techniques have been suggested to improve the quality of Web information retrieval results. They used a variety of approaches that depend on several data sources and employ advanced methods to find the best expansion keywords. However, most of these techniques failed to improve the retrieval performance of Web search engines and returned irrelevant information.

To cope with this limitation and solve the problem of query expansion in Web IR systems, we propose a new modelling of QE that aims to find the suitable expanded query instead of generating the best expansion keywords. In this new modelling, the best expanded query is selected from among a set of expanded query candidates. This set of expanded query candidates includes all possible combinations of keywords belonging to the pseudo-relevant documents.

Owing to the huge number of potential expanded query candidates, it is too complex to find the best one through conventional hard computing methods as they require incredibly large processing time, especially for large-scale data sources such as the Web. To overcome this drawback and find the appropriate expanded query within reasonable time, we consider the problem of query

expansion as a combinatorial optimization problem and apply swarm intelligence algorithms: Bat Algorithm, Firefly Algorithm and Accelerated Particle Swarm Optimization, to solve this issue.

To obtain better results, we first conduct a preliminary experiment to fix the BA, FA and APSO parameters. Then, we extensively evaluate our approach using MEDLINE dataset. The experimental results show that the proposed BA, FA and APSO for query expansion succeed to enhance the ranking of the relevant document and yields a significant improvement over the state-of-the-art in terms of precision and mean average precision.

Summary

Publications in international scientific journals:

1. Khennak, I., Drias, H.: An Accelerated PSO for Query Expansion in Web Information Retrieval: Application to Medical Dataset. Applied Intelligence. (Accepted on February 2017)

2. Khennak, I., Drias, H.: Bat-Inspired Algorithm Based Query Expansion for Medical Web Information Retrieval. Journal of Medical Systems, vol. 41, no. 2, pp. 34, 2017.

3. Khennak, I., Drias, H.: A Firefly Algorithm-based approach for Pseudo-Relevance Feedback: Application to Medical Database. Journal of Medical Systems, vol. 40, no. 11, pp. 240, 2016.

4. Khennak, I., Drias, H.: Strength Pareto Fitness Assignment for Pseudo-Relevance Feedback: Application to MEDLINE. Frontiers of Computer Science. (Accepted en September 2016)

5. Khennak, I., Drias, H.: Proximity-Based Good Turing Discounting and Kernel Functions for Pseudo-Relevance Feedback. International Journal of Information Retrieval Research, vol. 7, no. 3, 2017.

6. Khennak, I., Drias, H.: Bat Algorithm for Efficient Query Expansion: Application to MEDLINE. IFAC-PapersOnLine, vol. 47, no. 12, pp. 1791-1796, 2016.

Communications at international conferences:

1. Khennak, I., Drias, H., Kechid, S.: A New Modeling of Query Expansion Using an Effective Bat-Inspired Optimization Algorithm. Proceedings of the 8th IFAC Conference on Manufacturing Modelling, Management and Control, 28-30 Juin, 2016, Troyes, France.

2. Khennak, I., Drias, H.: Bat Algorithm for Efficient Query Expansion: Application to MEDLINE. Proceedings of the 4th World Conference on Information Systems and Technologies, 22-24 Mars, 2016, Racife, Brésil.

3. Khennak, I., Drias, H.: Strength Pareto Fitness Assignment for Generating Expansion Features. Proceedings of the 3rd World Conference on Information Systems and Technologies, 1-3 Avril, 2015, Azores, Portugal.

4. Khennak, I., Drias, H., Moustghanemi, H.: An Effective Term-Ranking Function for Query Expansion Based on Information Foraging Assessment. Proceedings of the 2nd International Conference on Mining Intelligence and Knowledge, 10-12 Décembre, 2014, Cork, Ireland.

5. Khennak, I., Drias, H., Moustghanemi, H.: A Novel Term-Term Similarity Score Based Information Foraging Assessment. Proceedings of the 1st International Internet of Things Summit. 27 Octobre, 2014, Rome, Italie.

6. Khennak, I., Drias, H.: A New Term-Term Similarity Measure for Selecting Expansion Features in Big Data. Proceedings of the International Conference on Advanced Networking, Distributed Systems and Applications, 17-19, Juin, 2014, Bejaïa, Algérie.

7. Khennak, I., Drias, H.: Classification Non Supervisée Floue des Termes Basée sur la Proximité pour les Systèmes de Recherche d'information. Proceedings of the 10th COnférence en Recherche d'Information et Applications, 3-5, Avril, 2013, Neuchâtel, suisse.

8. Khennak, I., Drias, H.: Term Proximity and Data Mining Techniques for Information Retrieval Systems. Proceedings of the 1st World Conference on Information Systems and Technologies, 27-30 Mars, 2013, Algrave, Portugal.

Communications in national conferences:

1. Khennak, I., Drias, H.: A Novel Term-Term Similarity Score Based Information Foraging Assessment. Conférence sur l'Ingénierie Informatique (C2i), Décembre, 2014, Alger, Algérie.

2. Khennak, I., Drias, H.: Un Système de Recherche d'Information Basé sur la Proximité des Termes. Conférence sur le Génie Electrique, Avril, 2013, Alger, Algérie.

Appendix A

Scientific production

TABLE A.1: Summary of scientific production (in numbers)

Publications in international scientific journals	6
Papers as "Book Chapters"	5
Communications at international conferences (abroad)	7
Communications at international conferences (in Algeria)	1
Communications in national conferences	2

A.1 International scientific journals:

#1

Publisher: Springer

Journal: Applied Intelligence

Indexing: JCR, SCOPUS, INSPEC, ACM, DBLP, ...

Volume: TBD

Issue: TBD

Pages: TBD

Year: TBD

Title: An Accelerated PSO for Query Expansion in Web Information Retrieval: Application to Medical Dataset

Abstract: Swarm intelligence algorithms are now among the most widely used soft computing techniques for optimization and computational intelligence. One recent swarm intelligence algorithm that has begun to receive more attention is Accelerated Particle Swarm Optimization (APSO). It is an enhanced version of PSO with global optimization capability, sufficient simplicity and high flexibility. In this paper, we propose the application of the APSO technique to efficiently solve the problem of Query Expansion (QE) in Web Information Retrieval (IR). Unlike prior studies, we introduce a new modelling of QE that aims to find the suitable expanded query from among a set of expanded query candidates. Nevertheless, due to the large number of potential expanded query candidates, it is extremely complex to produce the best one through conventional hard computing methods. Therefore, we propose to consider the problem of QE as a combinatorial optimization problem and address it with APSO. We thoroughly evaluate the proposed APSO for QE using MEDLINE, the world Web's largest medical library. We first conduct a preliminary experiment to tune the APSO parameters. Then, we compare the results to a recent swarm intelligence algorithm called Firefly Algorithm (FA). We also compare the results with three recently published methods for QE that involved Particle Swarm Optimization (PSO), Genetic Algorithm (GA) and Bat Algorithm (BA). The experimental analysis demonstrates that the proposed APSO for QE is very competitive and yields substantial improvement over the other methods in terms of retrieval effectiveness and computational complexity.

Link: TBD

Authors: Ilyes KHENNAK, Habiba DRIAS

#2

Publisher:	Springer
Journal:	Journal of Medical Systems
Indexing:	JCR, SCOPUS, INSPEC, ACM, DBLP, ...
Volume:	41
Issue:	2
Pages:	34
Year:	2017
Title:	Bat-Inspired Algorithm Based Query Expansion for Medical Web Information Retrieval
Abstract:	With the increasing amount of medical data available on the Web, looking for health information has become one of the most widely searched topics on the Internet. Patients and people of several backgrounds are now using Web search engines to acquire medical information, including information about a specific disease, medical treatment or professional advice. Nonetheless, due to a lack of medical knowledge, many laypeople have difficulties in forming appropriate queries to articulate their inquiries, which deem their search queries to be imprecise due the use of unclear keywords. The use of these ambiguous and vague queries to describe the patients' needs has resulted in a failure of Web search engines to retrieve accurate and relevant information. One of the most natural and promising method to overcome this drawback is Query Expansion. In this paper, an original approach based on Bat Algorithm is proposed to improve the retrieval effectiveness of query expansion in medical field. In contrast to the existing literature, the proposed approach uses Bat Algorithm to find the best expanded query among a set of expanded query candidates, while maintaining low computational complexity. Moreover, this new approach allows the determination of the length of the expanded query empirically. Numerical results on MEDLINE, the on-line medical information database, show that the proposed approach is more effective and efficient compared to the baseline.
Link:	https://dx.doi.org/10.1007/s10916-016-0668-1
Authors:	Ilyes KHENNAK, Habiba DRIAS

#3

Publisher:	Springer
Journal:	Journal of Medical Systems
Indexing:	JCR, SCOPUS, INSPEC, ACM, DBLP, ...
Volume:	40
Issue:	11
Pages:	240
Year:	2016
Title:	A Firefly Algorithm-based approach for Pseudo-Relevance Feedback: Application to Medical Database
Abstract:	The difficulty of disambiguating the sense of the incomplete and imprecise keywords that are extensively used in the search queries has caused the failure of search systems to retrieve the desired information. One of the most powerful and promising method to overcome this shortcoming and improve the performance of search engines is Query Expansion, whereby the user's original query is augmented by new keywords that best characterize the user's information needs and produce more useful query. In this paper, a new Firefly Algorithm-based approach is proposed to enhance the retrieval effectiveness of query expansion while maintaining low computational complexity. In contrast to the existing literature, the proposed approach uses a Firefly Algorithm to find the best expanded query among a set of expanded query candidates. Moreover, this new approach allows the determination of the length of the expanded query empirically. Experimental results on MEDLINE, the on-line medical information database, show that our proposed approach is more effective and efficient compared to the baseline.
Link:	https://dx.doi.org/10.1007/s10916-016-0603-5
Authors:	Ilyes KHENNAK, Habiba DRIAS

#4

Publisher:	Springer
Journal:	Frontiers of Computer Science
Indexing:	JCR, SCOPUS, INSPEC, ACM, DBLP, ...
Volume:	TBD
Issue:	TBD
Pages:	TBD
Year:	TBD
Title:	Strength Pareto Fitness Assignment for Pseudo-Relevance Feedback: Application to MEDLINE
Abstract:	Due to the growing use of unclear and imprecise keywords in characterizing the user's information need, it has become necessary to expand the original search query with additional words that best capture the actual user intent. Selecting the suitable terms to be used as additional words is largely dependent on the degree of relatedness between each candidate expansion word and the query keywords. In this work, we propose two criteria to evaluate the degree of relatedness between a candidate expansion word and the query keywords: (1) attribute more importance to terms occurring in the largest possible number of documents where the query keywords appear, (2) assign more importance to terms having a short distance with the query terms within documents. We also employ the strength Pareto fitness assignment in order to satisfy both criteria simultaneously. Our numerical experiments on MEDLINE, the online medical information database, show that the proposed approach significantly enhances the retrieval performance compared to the baseline.
Link:	TBD
Authors:	Ilyes KHENNAK, Habiba DRIAS

#5

Publisher: IGI Global

Journal: International Journal of Information Retrieval Research

Indexing: ACM, DBLP

Volume: 7

Issue: 3

Pages: TBD

Year: 2017

Title: Proximity-Based Good Turing Discounting and Kernel Functions for Pseudo-Relevance Feedback

Abstract: During the last few years, it has become abundantly clear that the technological advances in information technology have led to the dramatic proliferation of information on the web and this, in turn, has led to the appearance of new words in the Internet. Due to the difficulty of reaching the meanings of these new terms, which play an essential role in retrieving the desired information, it becomes necessary to give more importance to the sites and topics where these new words appear, or rather, to give value to the words that occur frequently with them. For this purpose, in this paper, we propose a new robust correlation measure that assesses the relatedness of words for pseudo-relevance feedback. It is based on the co-occurrence and closeness of terms, and aims to select the appropriate words that best capture the user information need. Extensive experiments have been conducted on the OHSUMED test collection and the results show that the proposed approach achieves a considerable performance improvement over the baseline.

Link: TBD

Authors: Ilyes KHENNAK, Habiba DRIAS

#6

Publisher: Elsevier

Journal: IFAC-PapersOnLine

Volume: 49

Issue: 12

Pages: 1791–1796

Year: 2016

Title: A New Modeling of Query Expansion Using an Effective Bat-Inspired Optimization Algorithm

Abstract: One of the most successful techniques to improve the retrieval effectiveness and overcome the shortcomings of search engines is Query Expansion (QE). Despite its effectiveness, QE still suffers from drawbacks that have limited its deployment as a standard component in search systems. Its major weakness is the computational cost, especially for large-scale data sources. To cope with this issue, we first propose in this paper, a judicious modeling of query expansion with a new and original metaheuristic namely, Bat-Inspired Approach to enhance the retrieval efficiency. Next, this approach is used to find both the best expansion keywords and the best relevant documents simultaneously unlike the previous works where these two tasks are performed sequentially. Our computational experiments undertaken on MEDLINE, the on-line medical database, show that our approach significantly enhances the retrieval efficiency over state-of-the-art methods.

Link: http://dx.doi.org/10.1016/j.ifacol.2016.07.842

Authors: Ilyes KHENNAK, Habiba DRIAS, Samir KECHID

A.2 Papers as "Book Chapters":

#1

Publisher:	Springer
Book Title:	New Advances in Information Systems and Technologies
Pages:	113–122
Year:	2016
Title:	Bat Algorithm for Efficient Query Expansion: Application to MEDLINE
Abstract:	Query expansion (QE) has long been suggested as an effective way to improve the retrieval effectiveness and overcome the shortcomings of search engines. Notwithstanding its performance, QE still suffers from limitations that have limited its deployment as a standard component in search systems. Its major drawback is the retrieval efficiency, especially for large-scale data sources. To overcome this issue, we first put forward a new modeling of query expansion with a new and original metaheuristic namely, Bat-Inspired Approach to improve the computational cost. Then, this approach is used to retrieve both the best expansion keywords and the best relevant documents simultaneously unlike the previous works where these two tasks are performed sequentially.
Link:	http://dx.doi.org/10.1007/978-3-319-31232-3_11
Authors:	Ilyes KHENNAK, Habiba DRIAS

#2

Publisher: Springer

Book Title: New Contributions in Information Systems and Technologies

Pages: 133–142

Year: 2015

Title: Strength Pareto Fitness Assignment for Generating Expansion Features

Abstract: Owing to the increasing use of ambiguous and imprecise words in expressing the user's information need, it has become necessary to expand the original query with additional terms that best capture the actual user intent. Selecting the appropriate words to be used as additional terms is mainly dependent on the degree of relatedness between a candidate expansion term and the query terms. In this paper, we propose two criteria to assess the degree of relatedness: (1) attribute more importance to terms occurring in the largest possible number of documents where the query keywords appear, (2) assign more importance to terms having a short distance with the query terms within documents. We employ the strength Pareto fitness assignment in order to satisfy both criteria simultaneously. Our computational experiments on OHSUMED test collection show that our approach significantly improves the retrieval performance compared to the baseline.

Link: http://dx.doi.org/10.1007/978-3-319-16486-1_13

Authors: Ilyes KHENNAK, Habiba DRIAS

#3

Publisher:	Springer
Book Title:	International Internet of Things Summit
Pages:	29–41
Year:	2014
Title:	A Novel Term-Term Similarity Score Based Information Foraging Assessment
Abstract:	The dramatic proliferation of information on the web and the tremendous growth in the number of files published and uploaded online each day have led to the appearance of new words in the Internet. Due to the difficulty of reaching the meanings of these new terms, which play a central role in retrieving the desired information, it becomes necessary to give more importance to the sites and topics where these new words appear, or rather, to give value to the words that occur frequently with them. For this aim, in this paper, we propose a novel term-term similarity score based on the co-occurrence and closeness of words for retrieval performance improvement. A novel efficiency/effectiveness measure based on the principle of optimal information forager is also proposed in order to assess the quality of the obtained results. Our experiments were performed using the OHSUMED test collection and show significant effectiveness enhancement over the state-of-the-art.
Link:	http://dx.doi.org/10.1007/978-3-319-19656-5_5
Authors:	Ilyes KHENNAK, Habiba DRIAS, Hadia MOSTEGHANEMI

#4

Publisher: Springer

Book Title: Mining Intelligence and Knowledge Exploration

Pages: 1–10

Year: 2014

Title: An Effective Term-Ranking Function for Query Expansion Based on Information Foraging Assessment

Abstract: With the exponential growth of information on the Internet and the significant increase in the number of pages published each day have led to the emergence of new words in the Internet. Owning to the difficulty of achieving the meaning of these new terms, it becomes important to give more weight to subjects and sites where these new words appear, or rather, to give value to the words that occur frequently with them. For this reason, in this work, we propose an effective term-ranking function for query expansion based on the co-occurrence and proximity of words for retrieval effectiveness enhancement. A novel efficiency/effectiveness measure based on the principle of optimal information forager is also proposed in order to evaluate the quality of the obtained results. Our experiments were conducted using the OHSUMED test collection and show significant performance improvement over the state-of-the-art.

Link: http://dx.doi.org/10.1007/978-3-319-13817-6_1

Authors: Ilyes KHENNAK, Habiba DRIAS, Hadia MOSTEGHANEMI

#5

Publisher: Springer

Book Title: Advances in Information Systems and Technologies

Pages: 477–486

Year: 2013

Title: Term proximity and Data Mining Techniques for Information Retrieval Systems

Abstract: Term clustering based on proximity measure is a strategy leading to efficiently yield documents relevance. Unlike the recent studies that investigated term proximity for improving matching function between the document and the query, in this work the whole process of information retrieval is thoroughly revised on both indexing and interrogation steps. Consequently, an Extended Inverted file is built by exploiting the term proximity concept and using data mining techniques. Then three interrogation approaches are proposed, the first one uses query expansion, the second one is based on the Extended Inverted file and the last one hybridizes retrieval methods. Experiments carried out on OHSUMED demonstrate the effectiveness and efficiency of our approaches compared to the traditional one.

Link: http://dx.doi.org/10.1007/978-3-642-36981-0_44

Authors: Ilyes KHENNAK, Habiba DRIAS

A.3 International conferences (abroad):

#1

Conference:	8th IFAC Conference on Manufacturing Modelling, Management and Control
Indexing:	/
Month:	June
Year:	2016
City:	Troyes
Country:	France
Title:	A New Modeling of Query Expansion Using an Effective Bat-Inspired Optimization Algorithm
Authors:	Ilyes KHENNAK, Habiba DRIAS, Samir KECHID

#2

Conference:	4th World Conference on Information Systems and Technologies
Indexing:	DBLP
Month:	March
Year:	2016
City:	Recife, Pernambuco
Country:	Brazil
Title:	Bat Algorithm for Efficient Query Expansion: Application to MEDLINE
Authors:	Ilyes KHENNAK, Habiba DRIAS

#3

Conference:	3rd World Conference on Information Systems and Technologies
Indexing:	DBLP
Month:	April
Year:	2015
City:	Azores
Country:	Portugal
Title:	Strength Pareto Fitness Assignment for Generating Expansion Features
Authors:	Ilyes KHENNAK, Habiba DRIAS

#4

Conference:	2nd International Conference on Mining Intelligence and Knowledge
Indexing:	DBLP
Month:	December
Year:	2014
City:	Cork
Country:	Ireland
Title:	An Effective Term-Ranking Function for Query Expansion Based on Information Foraging Assessment
Authors:	Ilyes KHENNAK, Habiba DRIAS, Hadia MOUSTGHANEMI

#5

Conference:	1st International Internet of Things Summit
Indexing:	DBLP
Month:	October
Year:	2014
City:	Roma
Country:	Italy
Title:	A Novel Term-Term Similarity Score Based Information Foraging Assessment
Authors:	Ilyes KHENNAK, Habiba DRIAS, Hadia MOUSTGHANEMI

#6

Conference:	1st World Conference on Information Systems and Technologies
Indexing:	DBLP
Month:	March
Year:	2013
City:	Algarve
Country:	Portugal
Title:	Term Proximity and Data Mining Techniques for Information Retrieval Systems
Authors:	Ilyes KHENNAK, Habiba DRIAS

#7

Conference:	10th COnférence en Recherche d'Information et Applications
Indexing:	DBLP
Month:	April
Year:	2013
City:	Neuchatel
Country:	Switzerland
Title:	Classification Non Supervisée Floue des Termes Basée sur la Proximité pour les Systèmes de Recherche d'information
Authors:	Ilyes KHENNAK, Habiba DRIAS

A.4 International conferences (in Algeria):

#1	
Conference:	International Conference on Advanced Networking, Distributed Systems and Applications
Publisher:	IEEE
Month:	June
Year:	2014
City:	Bejaia
Country:	Algeria
Title:	A New Term-Term Similarity Measure for Selecting Expansion Features in Big Data
Authors:	Ilyes KHENNAK, Habiba DRIAS

A.5 National conferences:

#1

Conference:	Conférence sur l'Ingénierie Informatique (C2i)
Month:	December
Year:	2014
City:	Algiers
Country:	Algeria
Title:	A Novel Term-Term Similarity Score Based Information Foraging Assessment
Authors:	Ilyes KHENNAK, Habiba DRIAS

#2

Conference:	Conférence sur le Génie Electrique
Month:	April
Year:	2013
City:	Algiers
Country:	Algeria
Title:	Un Système de Recherche d'Information Basé sur la Proximité des Termes
Authors:	Ilyes KHENNAK, Habiba DRIAS

Bibliography

[1] Ahmad, F., Kondrak, G.: Learning a spelling error model from search query logs. In: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, pp. 955–962. Association for Computational Linguistics (2005)

[2] Al Kabary, I., Schuldt, H.: Enhancing sketch-based sport video retrieval by suggesting relevant motion paths. In: Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1227–1230. ACM (2014)

[3] Anand, R., Kotov, A.: An empirical comparison of statistical term association graphs with dbpedia and conceptnet for query expansion. In: Proceedings of the 7th Forum for Information Retrieval Evaluation, pp. 27–30. ACM (2015)

[4] Baeza-Yates, R.: Information retrieval in the web: Beyond current search engines. International Journal of Approximate Reasoning **34**(2-3), 97–104 (2003)

[5] Banati, H., Bajaj, M.: Firefly based feature selection approach. IJCSI International Journal of Computer Science Issues **8**(4) (2011)

[6] Basu, B., Mahanti, G.K.: Fire fly and artificial bees colony algorithm for synthesis of scanned and broadside linear array antenna. Progress In Electromagnetics Research B **32**, 169–190 (2011)

[7] Bhatnagar, P., Pareek, N.: Genetic algorithm-based query expansion for improved information retrieval. In: Proceedings of the International Conference on Intelligent Computing, Communication and Devices, pp. 47–55. Springer (2015)

[8] Biancalana, C., Gasparetti, F., Micarelli, A., Sansonetti, G.: Social semantic query expansion. ACM Transactions on Intelligent Systems and Technology **4**(4), 60 (2013)

[9] Bindal, A.K., Sanyal, S.: Query optimization in context of pseudo relevant documents. In: Proceedings of the 3rd Italian Information Retrieval Workshop (2012)

[10] Bora, T.C., Coelho, L.d.S., Lebensztajn, L.: Bat-inspired optimization approach for the brushless dc wheel motor problem. IEEE Transactions on Magnetics **48**(2), 947–950 (2012)

[11] Brandao, W.C.: Exploiting entities for query expansion. ACM SIGIR Forum **48**(1), 43–43 (2014)

[12] Carpineto, C., Romano, G.: A survey of automatic query expansion in information retrieval. ACM Computing Surveys **44**(1), 1–50 (2012)

[13] Chatterjee, A., Mahanti, G.K., Chatterjee, A.: Design of a fully digital controlled reconfigurable switched beam concentric ring array antenna using firefly and particle swarm optimization algorithm. Progress In Electromagnetics Research B **36**, 113–131 (2012)

[14] Chen, Q., Li, M., Zhou, M.: Improving query spelling correction using web search results. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 181–189. Association for Computational Linguistics (2007)

[15] Curé, O.C., Maurer, H., Shah, N.H., Le Pendu, P.: A formal concept analysis and semantic query expansion cooperation to refine health outcomes of interest. BMC Medical Informatics and Decision Making **15**(1), 1 (2015)

[16] Eisenstein, J., Xing, E.P., Smith, N.A., O'Connor, B.: Mapping the geographical diffusion of new words. Tech. rep. (2012)

[17] El Ghali, B., El Qadi, A.: Context-aware query expansion method using language models and latent semantic analyses. Knowledge and Information Systems pp. 1–12 (2016)

[18] El Ghali, B., El Qadi, A., Ouadou, M., Aboutajdine, D.: Context-based query expansion method for short queries using latent semantic analyses. In: Proceedings of the 3rd International Conference on Networked Systems, pp. 468–473. Springer (2015)

[19] Gandomi, A.H., Yang, X.S., Alavi, A.H.: Mixed variable structural optimization using firefly algorithm. Computers & Structures **89**(23), 2325–2336 (2011)

[20] Gandomi, A.H., Yun, G.J., Yang, X.S., Talatahari, S.: Chaos-enhanced accelerated particle swarm optimization. Communications in Nonlinear Science and Numerical Simulation **18**(2), 327–340 (2013)

[21] Gao, J., Xu, G., Xu, J.: Query expansion using path-constrained random walks. In: Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 563–572. ACM (2013)

[22] Geng, B., Zhou, F., Qu, J., Zhang, B.W., Cui, X.P., Yin, X.C.: Social book search with pseudo-relevance feedback. In: Proceedings of the 21st International Conference on Neural Information Processing, pp. 203–211. Springer (2014)

[23] Hafiz, F., Abdennour, A.: Particle swarm algorithm variants for the quadratic assignment problems-a probabilistic learning approach. Expert Systems with Applications **44**, 413–431 (2016)

[24] Henzinger, M.R., Motwani, R., Silverstein, C.: Challenges in web search engines. In: ACM SIGIR Forum, vol. 36, pp. 11–22 (2002)

[25] Hitzler, P., Krotzsch, M., Rudolph, S.: Foundations of Semantic Web Technologies. CRC Press (2009)

[26] Horng, M.H.: Vector quantization using the firefly algorithm for image compression. Expert Systems with Applications **39**(1), 1078–1091 (2012)

[27] Imanirad, R., Yang, X.S., Yeomans, J.S.: Modelling-to-generate-alternatives via the firefly algorithm. Journal of Applied Operational Research **5**(1), 14–21 (2013)

[28] Jain, A., Mittal, K., Tayal, D.K.: Automatically incorporating context meaning for query expansion using graph connectivity measures. Progress in Artificial Intelligence **2**(2-3), 129–139 (2014)

[29] Jain, H., Thao, C., Zhao, H.: Enhancing electronic medical record retrieval through semantic query expansion. Information Systems and e-Business Management **10**(2), 165–181 (2012)

[30] Jalali, V., Borujerdi, M.R.M.: Information retrieval with concept-based pseudo-relevance feedback in medline. Knowledge and Information Systems **29**(1), 237–248 (2011)

[31] Jati, G.K., et al.: Evolutionary discrete firefly algorithm for travelling salesman problem. Springer (2011)

[32] Jiji, G.W., DuraiRaj, P.J.: Content-based image retrieval techniques for the analysis of dermatological lesions using particle swarm optimization technique. Applied Soft Computing **30**, 650–662 (2015)

[33] Jung, J.J.: Cross-lingual query expansion in multilingual folksonomies: a case study on flickr. Knowledge-Based Systems **42**, 60–67 (2013)

[34] Jurafsky, D., Martin, J.H.: Speech and Language Processing. Pearson (2014)

[35] Kassim, J.M., Rahmany, M.: Introduction to semantic search engine. In: Proceedings of the International Conference on Electrical Engineering and Informatics, pp. 380–386 (2009)

[36] Kazemzadeh Azad, S.: Optimum design of structures using an improved firefly algorithm. Iran University of Science & Technology **1**(2), 327–340 (2011)

[37] Kenndy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of 1995 IEEE International Conference on Neural Networks, pp. 1942–1948 (1995)

[38] Kennedy, J.: Particle swarm optimization. In: Encyclopedia of Machine Learning, pp. 760–766. Springer (2011)

[39] Kennedy, J., Kennedy, J.F., Eberhart, R.C., Shi, Y.: Swarm Intelligence. Morgan Kaufmann (2001)

[40] Khan, K., Sahai, A.: A comparison of ba, ga, pso, bp and lm for training feed forward neural networks in e-learning context. International Journal of Intelligent Systems and Applications **4**(7), 23 (2012)

[41] Khan, K., Sahai, A., Campus, A.: A fuzzy c-means bi-sonar-based metaheuristic optomization algorithm. International Journal of Interactive Multimedia and Artificial Intelligence **1**(7), 26–32 (2012)

[42] Khennak, I., Drias, H.: Bat algorithm for efficient query expansion: Application to medline. In: Proceedings of the 4th World Conference on Information Systems and Technologies, pp. 113–122. Springer (2016)

[43] Komarasamy, G., Wahi, A.: An optimized k-means clustering technique using bat algorithm. European Journal of Scientific Research **84**(2), 26–273 (2012)

[44] Leturia, I., Gurrutxaga, A., Areta, N., Alegria, I., Ezeiza, A.: Morphological query expansion and language-filtering words for improving basque web retrieval. Language Resources and Evaluation **47**(2), 425–448 (2013)

[45] Leung, C.H., Li, Y., Milani, A., Franzoni, V.: Collective evolutionary concept distance based query expansion for effective web document retrieval.

In: Proceedings of the 13th International Conference on Computational Science and its Applications, pp. 657–672. Springer (2013)

[46] Levene, M.: An Introduction to Search Engines and Web Navigation. John Wiley & Sons (2011)

[47] Lewandowski, D.: Web searching, search engines and information retrieval. Information Services & Use **25**(3, 4), 137–147 (2005)

[48] Li, Q., Tian, M., Liu, J., Sun, J.: An implicit relevance feedback method for cbir with real-time eye tracking. Multimedia Tools and Applications **75**(5), 2595–2611 (2016)

[49] Li, R., Hao, L., Zhao, X., Zhang, P., Song, D., Hou, Y.: A query expansion approach using entity distribution based on markov random fields. In: Proceedings of the 11th Asia Information Retrieval Societies Conference, pp. 387–393. Springer (2015)

[50] Ling, S.H., Chan, K.Y., Leung, F.H.F., Jiang, F., Nguyen, H.: Quality and robustness improvement for real world industrial systems using a fuzzy particle swarm optimization. Engineering Applications of Artificial Intelligence **47**, 68–80 (2016)

[51] Lu, Z., Kim, W., Wilbur, W.J.: Evaluation of query expansion using mesh in pubmed. Information Retrieval **12**(1), 69–80 (2009)

[52] Lv, C., Qiang, R., Fan, F., Yang, J.: Knowledge-based query expansion in real-time microblog search. In: Proceedings of the 11th Asia Information Retrieval Societies Conference, pp. 43–55. Springer (2015)

[53] Manning, C.D., Raghavan, P., Schütze, H., et al.: Introduction to Information Retrieval, vol. 1. Cambridge University Press Cambridge (2008)

[54] Miyanishi, T., Seki, K., Uehara, K.: Improving pseudo-relevance feedback via tweet selection. In: Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, pp. 439–448. ACM (2013)

[55] Moradi, P., Gholampour, M.: A hybrid particle swarm optimization for feature subset selection by integrating a novel local search strategy. Applied Soft Computing **43**, 117–130 (2016)

[56] Musikapun, P., Pongcharoen, P.: Solving multi-stage multi-machine multi-product scheduling problem using bat algorithm. In: Proceedings of the 2nd International Conference on Management and Artificial Intelligence, pp. 98–102 (2012)

[57] Oh, H.S., Jung, Y.: Cluster-based query expansion using external collections in medical information retrieval. Journal of Biomedical Informatics **58**, 70–79 (2015)

[58] Park, J.H., Croft, W.B.: Using key concepts in a translation model for retrieval. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 927–930. ACM (2015)

[59] Ramesh, B., Mohan, V.C.J., Reddy, V.V.: Application of bat algorithm for combined economic load and emission dispatch. International Journal of Electricl Engineering and Telecommunications **2**(1), 1–9 (2013)

[60] Reddy, V.U., Manoj, A.: Optimal capacitor placement for loss reduction in distribution systems using bat algorithm. IOSR Journal of Engineering **2**(10), 23–27 (2012)

[61] Ren, C., An, N., Wang, J., Li, L., Hu, B., Shang, D.: Optimal parameters selection for bp neural network based on particle swarm optimization: A case study of wind speed forecasting. Knowledge-Based Systems **56**, 226–239 (2014)

[62] Robertson, S., Zaragoza, H.: The probabilistic relevance framework: Bm25 and beyond. Foundations and Trends in Information Retrieval **3**(4), 333–389 (2009)

[63] Robertson, S.E., Jones, K.S.: Relevance weighting of search terms. Journal of the American Society for Information Science **27**(3), 129–146 (1976)

[64] Rocchio, J.J.: Relevance feedback in information retrieval. The SMART Retrieval System - Experiments in Automatic Document Processing pp. 313–323 (1971)

[65] Salehi, S., Selamat, A., Mashinchi, M.R., Fujita, H.: The synergistic combination of particle swarm optimization and fuzzy sets to design granular classifier. Knowledge-Based Systems **76**, 200–218 (2015)

[66] Saraiva, P.C., Cavalcanti, J.M., de Moura, E.S., Gonçalves, M.A., Torres, R.d.S.: A multimodal query expansion based on genetic programming for visually-oriented e-commerce applications. Information Processing & Management **52**(5), 783–800 (2016)

[67] Sarwar, S.M., Abedin, M.A., Ullah, A., Al Mamun, A.: Personalized query expansion for web search using social keywords. In: Proceedings of the 15th International Conference on Information Integration and Web-Based Applications and Services, p. 610. ACM (2013)

[68] Sayadi, M., Ramezanian, R., Ghaffari-Nasab, N.: A discrete firefly metaheuristic with local search for makespan minimization in permutation flow shop scheduling problems. International Journal of Industrial Engineering Computations **1**(1), 1–10 (2010)

[69] Senthilnath, J., Omkar, S., Mani, V.: Clustering using firefly algorithm: performance study. Swarm and Evolutionary Computation **1**(3), 164–171 (2011)

[70] Singh, J., Sharan, A.: Context window based co-occurrence approach for improving feedback based query expansion in information retrieval. International Journal of Information Retrieval Research **5**(4), 31–45 (2015)

[71] Singh, J., Sharan, A.: A new fuzzy logic-based query expansion model for efficient information retrieval using relevance feedback approach. Neural Computing and Applications pp. 1–24 (2016)

[72] Subramaniam, L.V., Roy, S., Faruquie, T.A., Negi, S.: A survey of types of text noise and techniques to handle noisy text. In: Proceedings of The 3rd

Workshop on Analytics for Noisy Unstructured Text Data, pp. 115–122. ACM (2009)

[73] Sun, H.M.: A study of the features of internet english from the linguistic perspective. Studies in Literature and Language **1**(7), 98 (2010)

[74] Swarnkar, K.K.: Economic load dispatch problem with reduce power losses using firefly algorithm. Journal of Advanced Computer Science & Technology **1**(2), 42–56 (2012)

[75] Tamiru, A., Hashim, F.: Application of bat algorithm and fuzzy systems to model exergy changes in a gas turbine. In: Artificial Intelligence, Evolutionary Computing and Metaheuristics, pp. 685–719 (2013)

[76] Tang, R., Fong, S., Yang, X.S., Deb, S.: Integrating nature-inspired optimization algorithms to k-means clustering. In: Proceedings of the 7th International Conference on Digital Information Management, pp. 116–123. IEEE (2012)

[77] Tsai, P.W., Pan, J.S., Liao, B.Y., Tsai, M.J., Istanda, V.: Bat algorithm inspired algorithm for solving numerical optimization problems. In: Applied Mechanics and Materials, vol. 148, pp. 134–137 (2012)

[78] Williams, H.E., Zobel, J.: Searchable words on the web. International Journal on Digital Libraries **5**(2), 99–105 (2005)

[79] Yang, X.S.: Nature-Inspired Metaheuristic Algorithms. Luniver Press (2008)

[80] Yang, X.S.: Firefly algorithms for multimodal optimization. In: Proceedings of the 5th International Conference on Stochastic Algorithms: Foundations and Applications, pp. 169–178 (2009)

[81] Yang, X.S.: Engineering Optimization: an Introduction with Metaheuristic Applications. John Wiley & Sons (2010)

[82] Yang, X.S.: Nature-Inspired Metaheuristic Algorithms: Second Edition. Luniver Press (2010)

[83] Yang, X.S.: A new metaheuristic bat-inspired algorithm. In: Nature Inspired Cooperative Strategies for Optimization, pp. 65–74 (2010)

[84] Yang, X.S.: Nature-Inspired Optimization Algorithms. Elsevier (2014)

[85] Yang, X.S., Hosseini, S.S.S., Gandomi, A.H.: Firefly algorithm for solving non-convex economic dispatch problems with valve loading effect. Applied soft computing **12**(3), 1180–1186 (2012)

[86] Yang, X.S., Karamanoglu, M., Fong, S.: Bat algorithm for topology optimization in microelectronic applications. In: Proceedings of the 1st International Conference on Future Generation Communication Technologies, pp. 150–155 (2012)

[87] Ye, Z., Huang, J.X.: A simple term frequency transformation model for effective pseudo relevance feedback. In: Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 323–332. ACM (2014)

[88] Yousif, A., Abdullah, A.H., Nor, S.M.: Scheduling jobs on grid computing using firefly algorithm (2011)

[89] Zhang, J.W., Wang, G.G.: Image matching using a bat algorithm with mutation. In: Applied Mechanics and Materials, vol. 203, pp. 88–93 (2012)

[90] Zhang, R., Song, S., Wu, C.: A two-stage hybrid particle swarm optimization algorithm for the stochastic job shop scheduling problem. Knowledge-Based Systems **27**, 393–406 (2012)

[91] Zhang, Y., Wu, L.: A novel method for rigid image registration based on firefly algorithm. International Journal of Research and Reviews in Soft and Intelligent Computing (IJRRSIC) **2**(2) (2012)

[92] Zhou, D., Lawless, S., Wade, V.: Improving search via personalized query expansion using social media. Information Retrieval **15**(3-4), 218–242 (2012)