Learning a Ranking Function for Information Retrieval using HybridABC

Supervisor: Dilruba Showkat

Co-Supervisor- Iffat Anjum

Conducted By:

S.M Saif Newaz-10201028

Maliha Anjum Pieta-09201009

Mehreen Ahmed- 11201010

Declaration

We, hereby declare that this thesis is based on the results found by ourselves. Materials of work found by other researcher are mentioned by reference. This Thesis, neither in whole or in part, has been previously submitted for any degree.

Signature of CO-Supervisor

_____

Iffat Anjum

Signature of Author

_____

S.M Saif Newaz

Signature of Author

_____

Maliha Anjum Pieta

Signature of Author

_____

Mehreen Ahmed

Acknowledgement

We want to express our gratitude to Almighty Allah (SWT) who gave us the opportunity, capability, determination and strength to work successfully on our thesis.

This research work was suggested by Ms. Dilruba Showkat, Lecturer, CSE, BRAC University, as a thesis of under graduation program. This thesis work is done by of Maliha Anjum Pieta, S.M Saif Newaz, Mehreen Ahmed, students of the CSE department of BRAC University. The document has been prepared as an effort to represent the knowledge obtained by us during these months of education on the field of bioinformatics.

We would like express sincere gratitude to our previous supervisor, Ms. Dilruba Showkat for her constant supervision, guidance and constant inspiration in completing our work.

We also want to acknowledge our new supervisor Ms. Iffat Anjum, who has consistently shown her interest in our work and supported us to fulfill this work. Thanks to Ms Dilruba Showkat and Ms Iffat Anjum, CSE Department, BRAC University for giving us the chance to work under their supervision.

Abstract

In this paper we propose a ranking algorithm, HybridABC that is built on swarm based algorithm. In our proposed HybridABC algorithm we merged Artificial Bee Colony (ABC) algorithm with Differential Evolution (DE) algorithm. The ABC is a swarm-based meta-heuristic algorithm inspired by the intelligent foraging pattern of bees and Differential Evolution is a population-based stochastic search technique. The proposed implementation of ABC has been tested using the LETOR dataset, which is a standard benchmark dataset for evaluating ranking functions. Our results display that our proposed HybridABC can compete and in many cases more efficient than other state-of-the-art algorithm proposed in ranking web pages based on Genetic Algorithm (GA).

Abbreviation

ABC= Artificial Bee colony

DE= Differential Evolution

MAP= Mean Average precision

NDCG=Normalized discounted cumulative gain

HybridABC= Hybrid ABC algorithm (proposed)

Table of Contents

1. INTRODUCTION

Genetic algorithms and swarm intelligence based algorithms are nature inspired algorithms which are implemented for fact that many real-world optimization problems have become increasingly huge, complex and dynamic. The size and complexity of the problems nowadays require the extension of techniques and solutions whose efficiency is measured by their capacity to find acceptable results within a logical and reasonable amount of time. Swarm intelligence impersonates the intelligent behavior of sets of individuals with limited intellectual capacity. The exact same principles of swarm intelligence in nature can be used in optimization algorithms. Ant colony, a colony of bees or an immune system are typical models of a swarm system. These algorithms could be sorted into various groups depending on the criteria being considered, such as, deterministic, iterative based, population based, stochastic, etc. An artificial swarm consists of a group of cooperative autonomous individuals, called agents. These agents satisfy their own purposes through cooperation with other agents. Communication and coordination are usually limited to a certain range, cooperation between agents only occurs locally. Optimization is the procedure of searching the best way to use available resources, while at the same time not violating any of the required conditions. Users generally demand that a practical minimization technique should fulfill several requirements like -

- Ability to handle different type of problems
- Ease of use with few control variables
- Good convergence mechanism to the global minimum in consecutive independent trials.

Artificial bee colony (ABC) algorithm is a biological-inspired population-based algorithm, recently proposed by D. Karaboga, which impersonates the foraging behavior of honey bee swarm [1]. Performance of ABC algorithm has been verified to be competitive to other

population-based algorithms with a benefit of simplicity and having less control parameters. ABC has been applied to solve different real world problems such as multilevel thresh holding and optimization problems, such as machining process, scheduling, structural design problem and power electric. ABC employed in this work as a hybrid with differential evolution (DE) in order to increase exploitation, the ability of searching near a candidate solution. DE is a very efficient evolutionary algorithm proposed by Storn and Price, whose performance has been improved and broadly accepted in many areas. The hybridization in this work increases ABC's exploitation without additional algorithmic parameters.

The exponential expansion of data on the World Wide Web is a challenge for the Search Engines. User needs to use different information retrieval tools for their desired information. Search engine is a tool used to fetch required information's from the World Wide Web. The structural design of search engine is shown in fig.2, consisting of three main elements: Crawler, Indexer and Ranking mechanism. Crawler negotiates the web and collects web pages from the web. Collected web pages are sent to index module; indexer creates and maintains the index of the pages. When a user posts a query in the interface of the search engine, query processor component match is the query keywords with the index and returns the URLs of the pages to the user. Ranking method is applied before showing results to the user. Page ranking was first introduced to rank web pages based on their significance on the web. It is a fundamental requirement of search engines to make the search results up-to-date and also insure the arrival speed.

In any information retrieval system ranking plays a main Role. Most of the Search engines return million of pages for a posted query, it is highly impossible or unfeasible for a user to observe all the returned results, here ranking is very helpful. Based on content and connectivity, ranking is divided into two categories. Content based ranking is depends on content of web page and connectivity-based ranking depends on link analysis technique. There are two famous link analysis techniques:

i) Page Rank Algorithm

ii) HITS Algorithm.

Page ranking algorithms are used by the search engines to display the search results by considering the importance, relevance, and content point. Web mining methods are employed by the search engines to take out relevant documents from the web database documents and provide the essential and required information to the users. If the search results are not presented according to the user interest then the search engine will lose its popularity. So the ranking algorithms become very important. Most vital link analysis algorithm is "PAGERANK" developed by Google. If the contents of any web page are frequently updated the owner with most significant data, definitely the user will heuristically gets attracted towards that web page and this makes the web page to get more interests than his competitors. On the other hand this is not possible with Page Rank algorithm as the referential theory only gives URL irrespective of the content. As a reason though the content is improved, it is not represented properly, Page Rank algorithm considers only URL's but not updated of contents. This article tries to deal with the above mentioned disadvantages by proposed hybrid ABC Approach which is consist of crossover of original ABC and mutation of DE or Deferential Evaluation. This proposed approach calculates User interest, Growth Analysis rate, and Total site linking. This algorithm eventually gives more relevant information for the query posted by the user when compared to the existing Page Rank algorithm. The results are proved to be encouraging. The proposed algorithm can be adopted by any Search Engine.

Figure 1 shows basic search engine architecture:

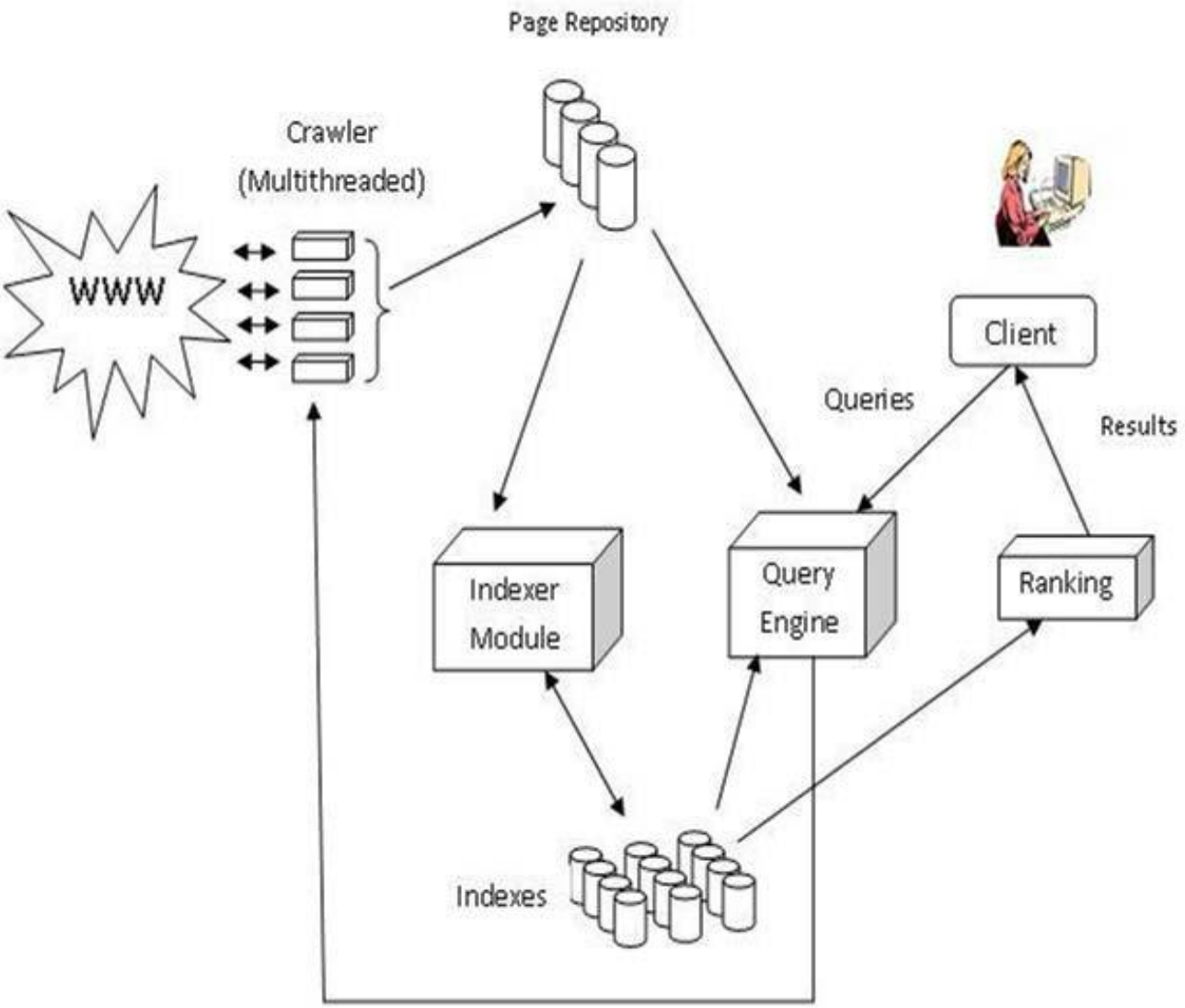

Figure 1: Basic Search engine architecture [4]

1.1 Objective

Learning a ranking function is important for numerous tasks such as information retrieval (IR), question answering, product recommendation etc. For example, IR for a web search engine is required to rank and return a set of documents relevant to a query based user. We propose HybridABC, a ranking technique that utilizes artificial bee colony (ABC) along with differential evaluation (DE) to learn a ranking function to position list of documents retrieved by a web search engine.

ABC has its own mutation but unlike other genetic algorithm (GA) it doesn't have crossover. So in our proposal, we have ranked web search engine by implementing crossover into ABC that is taken from DE which was used previously for ranking function along with ABC's own mutation, which is called Hybrid Artificial Bee Colony.

We have evaluated the proposed method on LETOR dataset- a benchmark dataset developed by Microsoft research to systematically evaluate different rank learning methods.

1.2 Motivation

Ranking pages for a search engine has always been important for proper web application. Giving the best result against a user query is the main purpose of a search engine. Previously different Genetic algorithms have been used in ranking web pages.

We tried to contribute in ranking web pages using a different algorithm which has never been used to rank web pages or in the sector information retrieval. The importance of ranking for information retrieval is the main motivation for us to work in this sector and try to improve ranking system.

1.3 Thesis Outline

- Section 1 deals with Introduction and Motivation of this project.
- In section 2 we explained about the background research which we did in order to make our proposed algorithm.
- We have included the learning rank in section 3
- In section 4 we explained our proposed algorithm.
- Section 5 deals with the experiments and results of our research.
- Features analysis is discussed in section 6
- In section 7 we have offered some concluding words on the research and our future plan of work with this algorithm.

2. Background Research

2.1 Genetic Algorithm

Genetic algorithms (GAs) are efficient, adaptive, and robust their search and optimization processes, use guided random choice as a tool for guiding the searching process in very large, complex, and multimodal search spaces. GA is modeled on the principles of natural genetic systems, where the genetic information of each individual or potential solution is encoded in structures called *chromosomes*. They use some domain- or problem-dependent knowledge to direct the search to more promising areas; this is known as the *fitness function*. Each individual or chromosome has an associated fitness function, which indicates its degree of goodness with respect to the solution it represents. Various biologically inspired operators such as *selection*, *crossover*, and *mutation* are applied to the chromosomes to harvest potentially better solutions. Note that the classical gradient search techniques perform efficiently when the problems under consideration satisfy tight constraints. However, when the search space is discontinuous and/or huge in size, noisy, high dimensional, and multimodal, GA has been found to consistently outclass both the gradient descent method and various forms of random search. Genetic algorithms (GAs) are adaptive computational procedures modeled on the mechanics of natural genetic systems. They efficiently exploit historical information to speculate on new offspring with improved performance.

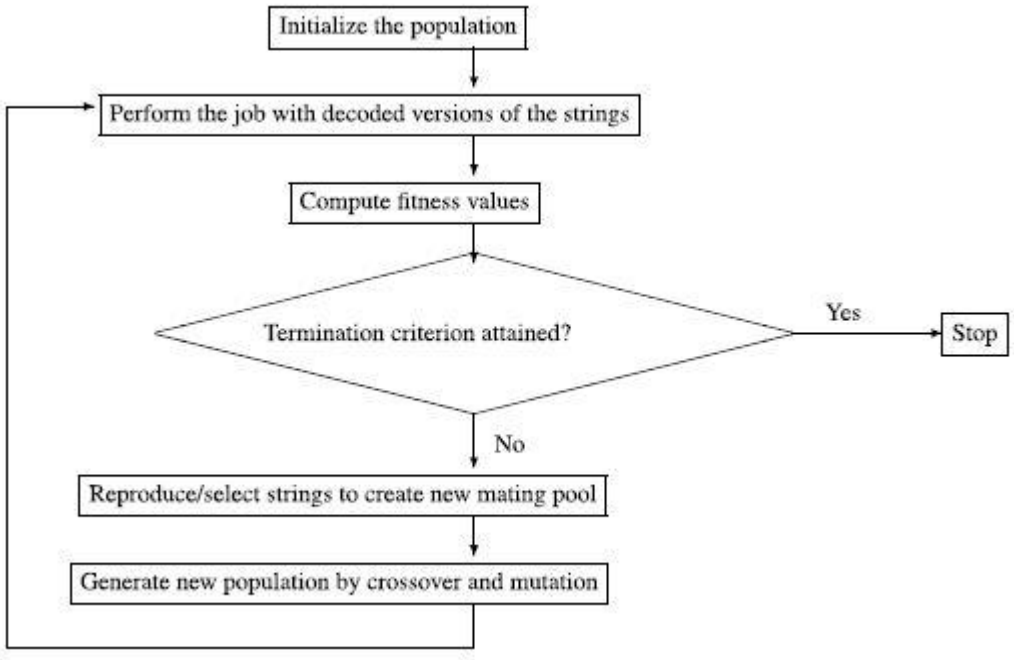

Figure 2: Basic Steps of GA [10]

As mentioned before, GA encodes the parameters of the search space in structures called a *chromosomes* (or *strings*). They execute iteratively on a set of chromosomes, called *population*, with three basic operators: *selection/reproduction*, *crossover*, and *mutation*. GA is different from most of the normal optimization and search procedures in four ways:

• GAs work with a coding of the parameter set, not with the parameters themselves.

• GAs work simultaneously with multiple points, and not with a single point.

• GAs search via sampling (blind search) using only the payoff information.

• GAs search using stochastic operators, not deterministic rules, to generate a new solutions.

2.2 Swarm Intelligence Based Algorithm

Swarm intelligence (SI) is the collective behavior of distributed, self-organized systems which are artificial or natural. It is comparatively a new category that deals with the study of self-organizing processes both in nature and in artificial systems. Researchers in etiology and animal behavior have proposed many models to explain interesting aspects of social insect behavior such as self-organization and shape-formation. The inspiration often comes from nature, especially biological systems. Recently, algorithms inspired by these models have been proposed to solve difficult computational problems. This concept was introduced by Gerardo Beni and Jing Wang in 1989, in the context of cellular robotic systems. This expression is used widely in artificial inelegancy. SI systems consist typically of a population of simple agents or bodies interacting locally with one another and with their environment. The agents follow very simple rules, and although there is no centralized control structure dictating how individual agents should behave, local, and to a certain degree random, interactions between such agents lead to the emergence of “intelligent" global behavior, unknown to the individual agents. Examples in natural systems of SI include ant colonies, bird flocking, animal herding, bacterial growth, fish schooling and microbial intelligence. Some human artifacts also fall into the domain of intelligence, notably some multi-robot systems and also certain computer programs that are written to tackle various problems occurring in the sector of optimization and data analysis.

2.3 Single Objective Optimization

Single objective would be the opposite of multi-objective optimization. In other words, standard optimizations with a single objective function. Multi-objective optimization means optimization with several competing objectives.

2.4 Particle Swarm Optimization

Particle swarm optimization (PSO) is a population-based stochastic approach for solving continuous and discrete optimization problems. It is a heuristic global optimization method put forward originally in 1995.and it is based on the research of bird and fish flock movement behavior. While searching for food, the birds are either scattered or go together before they locate the place where they can find the food. While the birds are searching for food from one place to another, there is always a bird that can smell the food very well, that is, the bird is perceptible of the place where the food can be found, having the better food resource information. Because they are transmitting the information, especially the good information at any time while searching the food from one place to another, conducted by the good information, the birds will eventually flock to the place where food can be found. As far as particle swam optimization algorithm is concerned, solution swam is compared to the bird swarm, the birds' moving from one place to another is equal to the development of the solution swarm, good information is equal to the most optimist solution, and the food resource is equal to the most optimist solution during the whole course. The most optimist solution can be worked out in particle swarm optimization algorithm by the cooperation of each individual.

2.5 Artificial Bee Colony

Artificial Bee colony (ABC) is an algorithm based on PSO. Tereshko model (a model of bee colony) consists of three necessary components: food sources, employed foragers and unemployed foragers, and defines two most important modes of the behavior: recruitment to a nectar source and abandonment of a source.

Food Sources: The value of a food source to an insect depends on many aspects including its proximity to the nest, richness or concentration of energy, and the simplicity of extracting this energy. The key point is to describe the profitability of a food source with a single quantity and to see how insects react to food sources with different values of this capacity, if they always are capable to select the best food source in a changing environment.

Employed Foragers: Employed foragers are associated with a particular food source which they are currently utilizing or are employed at. They carry with them information about this particular source, its distance and direction from the nest, and the profitability of the source. Employed foragers will share this information with a certain probability. The larger the profitability of a food source, the higher the probability the honeybee will do a waggle dance and share her information with her nest mates. However that employed foragers are only locally informed they know only of the food source they are currently exploiting and continue frequenting this food.

Unemployed Foragers: Unemployed foragers are looking for a food source to exploit. There are two types of unemployed foragers, scouts, who search the environment surrounding the nest (approximately up to a 14 km radius) in search of new food sources, and onlookers who wait in the nest and a food source through the information shared by employed foragers. The percentage of unemployed foragers who are scouts varies from 5% to as much as 30% depending on the influx of information into the nest. The mean number of scouts averaged over conditions is about 10%.Karaboga described this Tereshko model as Behavior of real

bees shown in Figure 3 to explain the connection of the above mentioned model.

Assume that there are two discovered food sources: A and B.
At the very beginning, a potential forager will start as unemployed forager. That bee will have no knowledge about the food sources around the nest.

There are two possible options for such a bee:
(1) It can be a scout and starts searching around the nest spontaneously for food due to some internal motivation or possible external clue (`S' in Figure 3).

(2) It can be a recruit after watching the waggle dances and starts searching for a food source (`R' in Figure 2).

After finding the food source, the bee utilizes its own capability to memorize the location and then immediately starts exploiting it. Hence, the bee will become an employed forager. The foraging bee takes a load of nectar from the source and returns to the hive, unloading the nectar into storage.

After unloading the food, the bee has the following options:

(1) It might become an uncommitted follower after abandoning the food source (UF).

(2) It might dance and then recruit nest mates before returning to the same food source (EF1).

(3) It might continue to forage at the food source without recruiting other bees (EF2). [3]

It is important to note that not all bees start foraging simultaneously. The experiments confirmed that new bees begin foraging at a rate proportional to the difference between the eventual total number of bees and the number presently foraging.

.

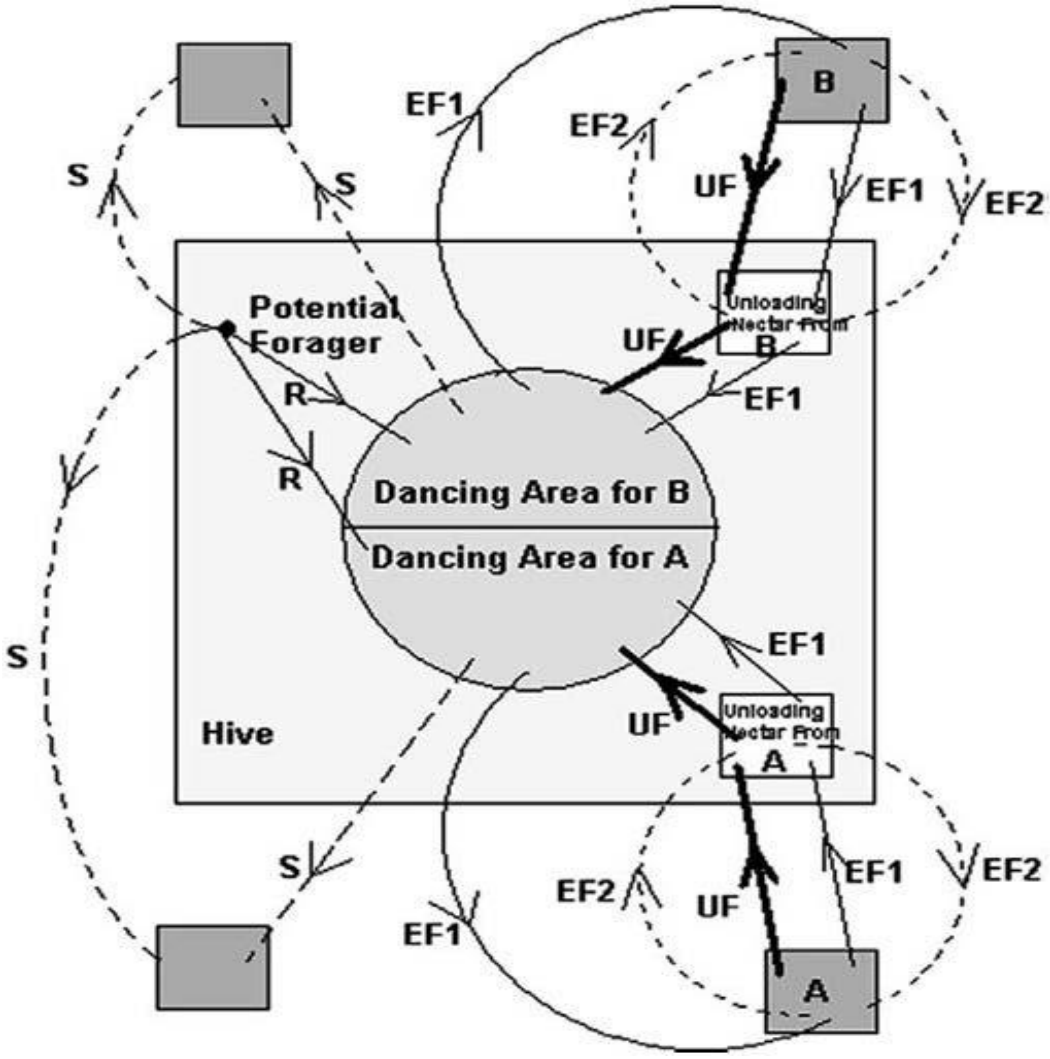

Figure 3: Behavior of the bees in Artificial Bee Colony [3]

2.5.1 Artificial Bee Colony Algorithm

In ABC algorithm, the colony of artificial bees contains three groups of bees: employed bees, onlookers and scouts. First half of the colony consists of the employed artificial bees and the second half includes the onlookers. For every food source, there is only one employed bee. In other words, the number of employed bees is equal to the number of food sources. The employed bee of an abandoned food source becomes a scout. The search carried out by the artificial bees can be summarized as follows:

(1) Employed bees determine a food source within the neighborhood of the food sourcing their memory.

(2) Employed bees share their information with onlookers within the hive and then the onlookers select one of the food sources.

(3) Onlookers select a food source located within their proximity.

(4) An employed bee of which the source has been abandoned becomes a scout and starts to search a new food source randomly.

The flowchart for Artificial Bee Colony is shown in figure 4 on the next page.

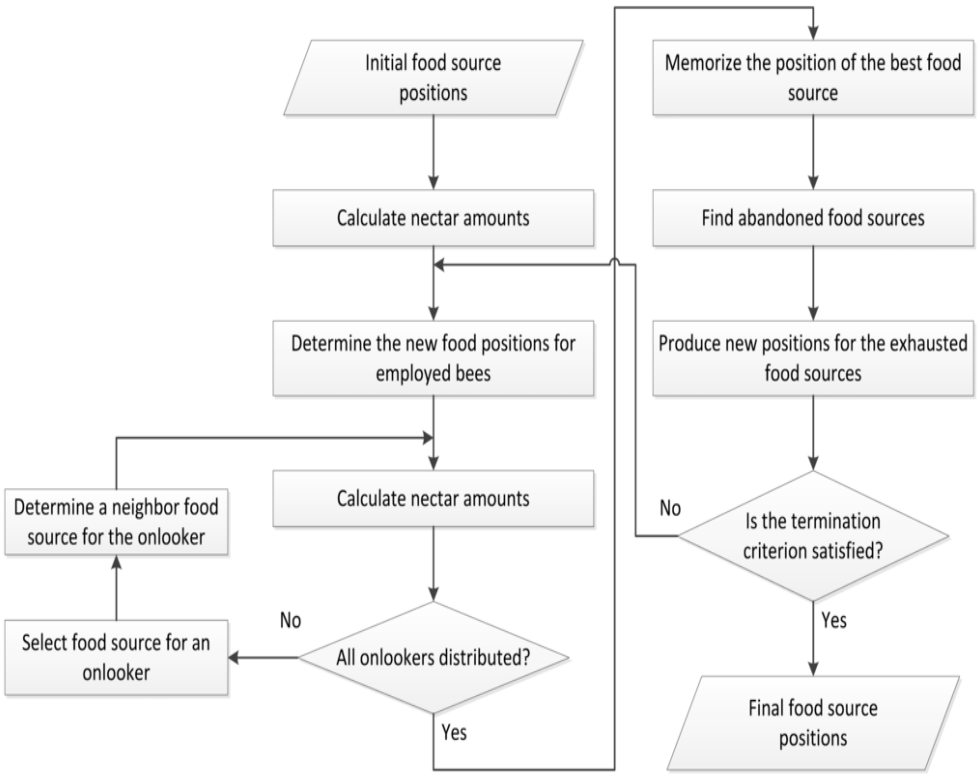

Figure 4: Flowchart of ABC Algorithm [15]

The main steps of the algorithm are given below:

1. INITIALIZE
2. REPEAT
3. Move the employed bees onto their food sources and determine their nectar amounts.
4. Move the onlookers onto the food sources and determine their nectar amounts.
5. Move the scouts for searching new food sources.
6. Memorize the best food source found so far.
7. UNTIL (requirements are met)

Each cycle of the search consists of three steps: moving the employed and onlooker bees onto the food sources and calculating their nectar amounts and determining the scout bees and then moving them randomly onto the possible food sources. A food source represents a possible solution to the problem to be optimized. The nectar amount of a food source corresponds to the quality of the solution represented by that food source.

2.6 Non-Dominated Sorting Genetic Algorithm ii (NSGA ii)

The original algorithm of NSGA has been evolved, confronting some issues and complications. NSGA presents problem in terms of computational complexities, a non-elitism approach and also for selecting an optimal parameter value for sharing parameter. NSGA has a computational complexity of $O(MN^3)$, where M represents the objectives and N represents the size of the population. So, for a large population size, NSGA becomes very taxing computationally. Due to the non-dominated sorting that happens after every computational round, we have to deal with a large complexity issue. Furthermore, the lack of elitism also constitutes a poor performance because for genetic algorithms, elitism can speed up the process and gain track on useful solutions once encountered. The third drawback being the specifying of a shared parameter, the goal is to enrich the priority of diversity in the results and hence eliminate the choosing of a parameter at all since the whole sharing concept means having to use a parameter that is shared. Keeping all these issues highlighted, the NSGA-II was developed. NSGA-II is a non-dominated multi-objective evolutionary genetic algorithm. It combines both crossover and mutation aspects thus can be implemented as a hybridized version algorithm. It has developed into newer version to reduce its time-complexity drawbacks and to improve its convergence rate. [10]

2.7 Differential Evolution

Differential evolution (DE), proposed by Storn and Price, is a simple yet powerful population-based stochastic search technique for solving global optimization problems. DE has been used successfully in numerous fields such as pattern recognition, communication, and mechanical engineering, to optimize non-convex, non-differentiable and multi-modal objective functions. DE has many attractive properties compared to other evolutionary algorithms such as, implementation simplicity, the small number of control parameters, fast convergence rate, and robust performance. DE has only a few control variables which remain fixed throughout the optimization process, which makes it easy to implement. Moreover, DE can be implemented in a parallel processing framework, which enables it to process a large number of training instances efficiently.[9] These properties of DE make it an ideal candidate for the current task of learning a ranking function for information retrieval, where we must optimize non-convex objective functions such as MAP and NDCG measures over large datasets. Next, we briefly outline the main steps in DE. For further details of DE and its comparison to other evolutionary computational approaches refer [2]. Without a loss of generality, we will consider the problem of maximizing a given objective function. (A similar approach can be followed for minimization.)

The Differential algorithm procedure can be broken down into three primary stages which are: mutation, recombination and selection, as can be seen in Figure 5.

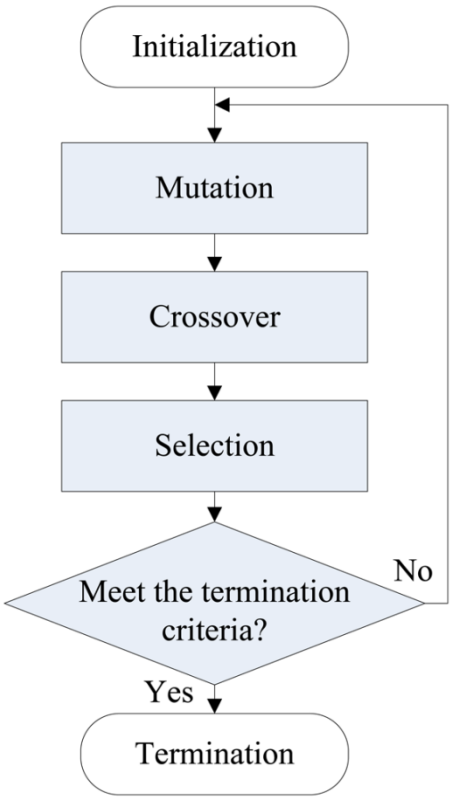

Figure 5: Flowchart of DE [13]

During the initialization, a population of vectors, N, is generated with defined upper and lower bounds for the parameters and the objective purpose also given. Then the target vectors are sequentially handled through the stages.

For the mutation procedure, either a random target vector is selected from the base vector or selects the optimum instance from population set of N. To derive mutant vector from basis vector the disparity between the selected pairs of instances are kept in basis vector: for the selected instance of vector, three other vectors are taken with separate indices and the weighed variation comparing that of the first two to the third is calculated and stored. For example for the selected vector $x_{i,G}$, the calculated result for mutation would be:

$$v_{i,G+1} = x_{r_1,G} + F(x_{r_2,G} - x_{r_3,G}) \quad \text{EQ1}$$

Where, r1, r2, r3 are indices and the solution $v_{i,G+1}$ is called the mutant vector or the donor

vector.

Next there is the recombination or crossover part in which the optimal solutions from the previous generation are worked with. This portion also enriches overall diversity of the population. From the elements of the target vector, a trial vector is formed and also using the elements of the mutant vector. Both mutant and target vectors substitute components to create a trial vector, $u_{i,G+1}$. Elements of mutant vector enter trial vector with a probability of CR (crossover constant). The equations for trial vector look like this:

$$u_{ji,G+1} = \begin{cases} v_{ji,G+1} & \text{IF } (randb(j) \leq CR) \text{ OR } (j = rnbr(i)), \\ x_{ji,G} & \text{IF } (randb(j) > CR) \text{ AND } (j \neq rnbr(i)). \end{cases} \quad \text{..... EQ2}$$

Where $u_{ji,G+1}$ is the trial vector, randb(j) represents j-th evaluation, rnbr(i) represents random integers within range [1,2,….D] and it is to make sure mutant vector does not equate with $x_{i,G}$ or target vector. And the Constant vector CR is a specified crossover constant which in this case is set to te range [-.05,.05].

In the final stage of selection, both target vector, $x_{i,G}$, and trial vector, $u_{ji,G+1}$,are checked and compared to select whichever has lowest function value so it can be part of next generation of population.

This whole process is on loop, and the new generation undergoes the three stages again until at the end the defined criterion is met with.

There are some significant characteristics involved in each step:

- Mutation enlarges the search space; it includes more genetic subjects into the population.
- Crossover probes into the newer regions of the search space.
- Selection reduces and contracts the problem to its specific solution. In other words where mutation increases diversity, the selection process decreases it.

3. Learning to Rank

In case of information retrieval, learning a rank problem is based on two phases. One is the training phase and in this scenario we learn a ranking function from a set of annotated training data. The other phase is the test phase where we apply the learned ranking function to rank a set of documents. These documents are retrieved by a search engine for a user query. In the training phase, a learning algorithm is presented with a collection of queries and their corresponding retrieved documents, the documents are assigned with some labels that indicate the relevance results of those documents to their corresponding query. The relevance results are assigned by human annotators. An annotator might annotate a set of documents by assigning some ranking score to each document depending on its relevance to the query. A higher ranking score indicates that a document with such a score is more relevant to the user query and this must be ranked at the top.

The objective of learning is to construct a ranking model (e.g. a ranking function) that achieves the best agreement with the ranking induced by the scores assigned by the human annotators. The agreement between a ranking algorithm and the set of documents from the human annotator can be measured using numerous rank evaluation metrics such as Mean Average Precision (MAP), Normalized Discounted Cumulative Gain (NDCG), Kendall's rank correlation coefficient and Spearman's rank correlation coefficient. Traditionally, both MAP and NDCG have been used in the information retrieval community to evaluate rankings in Web search engines because those measures are shown to be correlating well with the document relevance in the Web search settings.

After learning the ranking function where we used a set of training data for the result of a

query, now we will use it to rank a set of documents which will be retrieved for a user query. In the test phase let us name the set of queries Q = { q1….q[Q] }, here, Q is used to represent the number of elements which are in the set Q. Now, let D = { d1…d[D] },

where D is the set of documents in the query set (Q in this case). Now the datasets are created as a pair of query-document. Now, { q(i) , d(i) } ε Q X D this pair of documents are assigned with a label y{ q(i), d(i)} indicates the relationship between the query and the document. The judgment from the result can easily be a binary relevance which will indicate that if the document is relevant or non-relevant to the query. A query-document pair (q, d) is represented using a feature vector f(q; d) = $\omega\phi$(q, d)……..(1). Here ωrepresents the weight vector and ϕ represents the feature vector. In order to rank the documents retrieved from the query we compute {q(i),d(i)} each retrieved document d(i) using the equation, f(q,d) = $\omega\phi$(q, d). After obtaining the result we sort the documents in descending order. The equation f(q, d) = $\omega\phi$(q, d) is well known in previous work in this field.

4. Proposed Rank Learning Method: HybridABC

Motivated by the success of Artificial Bee Colony (ABC) algorithm in various sectors and Differential Evolution (DE) being used for a wide range of tasks, we propose HybridABC algorithm. This algorithm is a compilation of artificial bee colony algorithm merged with differential evolution algorithm. In this algorithm the artificial bee colony algorithm is used to select food source (i.e. Training dataset) and sort the food source. In stock ABC algorithm, the food source would be evaluated by the bees but in our proposed algorithm we here use the fantastic evolution method of differential evolution. The food source or in other words the input datasets which we used to determine the capability of our algorithm contain query pairs (q ,d) with the corresponding feature vectors f(q, d). Here in the employee bee phase the bees take the input sets in their memory and share the information with the onlooker bees. Now in normal ABC algorithm these onlooker bees will be selecting good food source and then ranking them using ABC algorithm's own greedy algorithm. But here, in our proposed HybridABC algorithm we do not use the traditional format of ABC algorithm and we use the evolution methods of DE here to evaluate the food source using two equations through which best fitness of the source is obtained. Then the food source is sent to the main fitness of the algorithm and then is memorized after checking for any better solution found by the scout bees. Then the result will include the higher food source. The scout bees however are some bees which transform from employed bees and search for new food source. After evaluating the food source the final output of Algorithm 1 is the parameter vector that maximizes the fitness function. In our case we use Mean Average Precision (MAP) and Normalized discounted cumulative gain (NDCG) as the fitness function.

ALGORITHM 1

Initialize: Generate the initial population z(i)=1,2,…,SN

Evaluate the fitness of the population,

Cycle=1;

REPEAT

FOR each employee bee{

Produce new solution

Calculate the values Fitness

Apply greedy selection process}

Calculate probability values p(i), for the solutions z(i),

FOR

Each onlooker bee{

Select distinct r1,r2,r3 randomly from z(i) based on p(i)

Produce new solution

Compute v(i);G+1 using EQ1

$$v_{i,G+1} = x_{r_1,G} + F(x_{r_2,G} - x_{r_3,G}) \quad \text{EQ1}$$

Compute u(i);G+1 using EQ2

$$u_{ji,G+1} = \begin{cases} v_{ji,G+1} & \text{IF } (randb(j) \le CR) \text{ OR } (j = rnbr(i)), \\ x_{ji,G} & \text{IF } (randb(j) > CR) \text{ AND } (j \ne rnbr(i)). \end{cases} \quad \text{EQ2}$$

If(u(i);G+1;z(i)) > (x(i); G+1;z(i))……..[where x is a weight vector]

Then

x(i);G+1=u(i);G+1

end if

end for

return f(q,d) = ω ϕ (q,d), where ω is the highest fitness value

Fitness=f }

end for

If an abandoned solution for the scout exists,

Replace it with new solution,

Memorize the best Fitness,

Cycle++;

Until Cycle==MCN.

5. Experiment

In this Section, we first describe the LETOR dataset and then we present the experimental results on this benchmark dataset.

5.1 Dataset

For the Hybrid ABC algorithm we intended to use a reliable and compatible dataset such as the LETOR (version 2.0) benchmark dataset.

LETOR (Learning to Rank for Information Retrieval) is a source for a group of standard datasets that are applicable depending on their setting or versions for instance supervised or list-wised ranking purposes, originally released by Microsoft Research Asia.

This dataset package is tested with our designed hybrid algorithm and some other algorithms for information retrieval, and the results are thus weighed against each other to analyze which algorithm succeeds. The LETOR 2.0 contains TD2003 and TD2004 datasets, the features of which are shown in table 1 below. From the TREC and OHSUMED collections all the related query-document pair elements were taken out as a part of the datasets (for information retrieval) and the TD2003 and TD2004 datasets represent the ones from years 2003 and 2004. The 2003 dataset consists of 50 queries whereas the 2004 dataset has 75 queries. The whole document compiled has well over a million documents including 11,164,829 hyperlinks as of the January 2002 listings. The query-document pair overlooks the binary judgments procedure that studies a document to reveal if it is relevant to the specific query or not.[8]

The query-document pair is set up using a 44-dimensional feature vector. An array of retrieved documents is ranked using the ranking heuristics, in the feature vector, that is typical for IR. LETOR has in its library both high-level and low-level features and also a combination of low-level features. To test the applicability of a web page, the hyperlink configuration helps to sort out. So a lot of features are analyzed and ranked using the hyperlink structure found in the datasets (e.g. HostRank, PageRank etc). After the values have been computed, the results must

be normalized in order to be able to compare between the values of feature from the retrieved documents for a given query, since at first it was not compatible.

The normalized value comes out as:

$$\phi_k(q_i, d_j) = \frac{\phi_k(q_i, d_j) - \min\{\phi_k(q_i, d_j)\}}{\max\{\phi_k(q_i, d_j)\} - \min\{\phi_k(q_i, d_j)\}}. \quad \text{.....Eq(3)}$$

Where d_j is the document in set, q_i is the query, k represents the k-th feature in the vector $\phi(q_i, d_j)$. It is noteworthy that the values of features extracted for documents retrieved for different queries are not comparable. Therefore, we first normalize the values of each feature across all documents retrieved for a particular query. Let us denote the set of documents Retrieved for query q_i by $D(q_i)$ and a document in this set by dj (i.e. d_j 2 $D(q_i)$). Moreover, let us denote the k-th feature in the feature vector -(q_i ; d_j) representing a query-document pair (q_i ; d_j) by -k(q_i ; d_j). Then, the normalized value of -k(q_i ; d_j) 2 [0; 1] is calculated from the equation above.

Category	Feature	No. of features
Content (low-level)	tf [1]	4
	idf [1]	4
	dl [1]	4
	tfidf [1]	4
Content (high-level)	BM25 [23]	4
	LMIR [33]	9
Hyperlink	PageRank [19]	1
	Topical PageRank [18]	1
	HITS [16]	2
	Topical HITS [18]	2
	HostRank [31]	1
Hybrid	Hyperlink-base relevance propagation [24]	6
	Sitemap-based relevance propagation [21]	2
Total		44

TABLE 1: TD2003, TD2004 dataset features

The contents extracted from various OHSUMED and TREC collections for the most part take care of the basic benchmark features of information retrieval (IR), along with classical properties (e.g. inverse document frequency, BM25 ,language models etc) , as well as features found under the SIGIR research.

The LETOR is boasted as being a standardized staple among other contemporary and latest ranking models with these characteristics and a compiled baseline results for studies carried out so far plus for the future. So it is very effectively an evaluation tool that helps analyze the contrasts between various methods when tested on a few common similar features.

5.2 Evolution Measures

We have to test our algorithm in order to see if our proposed Hybrid ABC algorithm can run the datasets and give relevant result against the queries. Now we used our dataset on our algorithm to get the maximum fitness . Now we have to compete it with the ranking induced by human annotator for a specific set of documents, which we also used in testing our algorithm. In order to do that, we have to use an evolution measure. Precision at position n (P @n), Mean Aver-age Precision (MAP), and normalized discounted cumulative gain(NDCG) are three widely used rank evaluation measures in the in-formation retrieval community. Now Both those evaluation measures are in the range [0,1], where a method that produces the exact ranking as in the gold standard achieves the score of 1. Next, we describe each of those evaluation measures in detail.

Precision at rank n (P @n) measure is defined as the proportion of the relevant documents among the top n-ranked documents.

$$P@n = \frac{\text{No. of relevant docs in top } n \text{ results}}{n}.$$

Average precision averages the P @n at over different n values to produce a single measure for a given query as follows,

$$AP = \frac{\sum_{n=1}^{N}(P@n \times rel(n))}{\text{No. of relevant docs for this query}}.$$

No. of relevant docs for this query Here, N is the number of retrieved documents, and rel(n) is a binary function that returns the value 1 if the nth ranked document is relevant to the query under consideration and 0 otherwise. Mean average precision (MAP) is computed as the average of AP over all queries in the dataset.

NDCG considers the reciprocal of the logarithm of the rank as-signed to relevant documents. For a ranked list of documents retrieved for a query, NDCGvalue at position n, NDCG@n, is computed as follows,

$$NDCG@n = Z_n \sum_{j=1}^{n} \frac{2^{r(j)} - 1}{\log(1+j)}.$$

Here, r(j) is the rating of the j -th document in the ranked list, and the normalization constant Zn is chosen such that a perfectly ranked list would obtain an NDCG@n score of 1. Specifically, it is given by,

$$Z_n = \frac{1}{\sum_{j=1}^{n} \frac{1}{\log(1+j)}}.$$

For the TD2003 and TD2004 datasets, we define two values of ratings 0 and 1 respectively corresponding to relevant and non-relevant documents in order to compute NDCG scores. In our evaluations, we report the average values taken over all the queries in a dataset as P @n and NDCG@n.

5.3 Parameter Settings

The parameters in our proposed ranking algorithm Hybrid ABC are set to the some certain values. The population is set to be 50, no of dimension 44 and random generation is set to be 10000. We measured the performance on the validation data provided inthe LETOR datasets. Each individual is represented by a 44 dimensional real-valued vector in which, each dimension corresponds to some feature found in the LETOR datasets. The initial population is generated randomly by selecting the parameter values from the range [1,1].

5.4 Evolution Procedure and Baselines

We have compared our proposed Hybrid ABC learning to rank algorithm, with some baselines Microsoft LETOR Dataset - TD2003 and TD2004 datasets that was proposed previously. Next, we have described each of those algorithms shortly.

BM25:BM25is a non-learning ranking function, which combines several statistics to compute a ranking score that reflects the relevancy of a document to a given query or request. Information such as the number of times the query occur in a document (i.e. term frequency), the number of documents that holds the query (i.e. document frequency), the length of the document in words, and the average length of a document (i.e. the average number of words contained in any document in the collection)are utilized by it. The Okapi information retrieval system had used BM25 successfully and it is popularly known as Okapi BM25. This baseline demonstrates the performance that we would obtain if we did not use any training data to learn a ranking function.

RankSVM: Ranking Support Vector Machine is an extension to the standard binary support vector classifier that performs ordinal regression. Particularly, RankSVM learns a large margin classifier, which minimizes the number of conflicting pairs between two sets of ranks. It is a pair wise learning algorithm that optimizes the rank evaluation measures indirectly such as the MAP, through minimizing the number of conflicting pairs between a human-made ranking and a system-made ranking.

RankBoost: RankBoost was proposed by Freund et al to combine multiple rankings using the AdaBoost algorithm. Rank-Boost works by combining multiple weak rank scores of a given set of training instances. The weak rank scores might be only weakly interrelated with the target (human assigned) ranking scores. RankBoost is able to learn an accurate ranking function by combining such a set of weak rank scores via boosting. The features exist in the LETOR datasets such as term-frequency; PageRank, BM25, etc. are used to create the weak rankings by the RankBoost algorithm.

SwamRank: This is a particle swam optimization (PSO)-based ranking algorithm and it attempts to learn a linear combination of various ranking functions in the form:

$$f(q,d) = \boldsymbol{w}^{\top}\boldsymbol{\phi}(q,d).$$

SwamRank optimizes the MAP for a given training dataset directly.

GpRank: Yeh et al proposed this and it is the genetic programming-based rank learning algorithm for information retrieval. Similar to SwamRank, GPRank directly optimizes the MAP for a given training dataset.

HybridABC: This is the Combination of Artificial Bee Colony (ABC) algorithm and Differential Evolution (DE) which we named HybridABC a rank learning algorithm proposed in this paper.

We conduct 5-fold cross-validation using the LETOR datasets by following the official

guidelines and data partitions as described in the LETOR project. We use three subsets for each foldas training data, one subset as validation data, and the other subset for testing. To report the performance of the ranking function learnt by HybridABC, we compute the evaluation measures MAP, P@n, and NDCG@n on the test. The reported performance in this paper is the average over the five folds.

5.5 Results

Tables 2 and 3 compare the performance of the proposed HybridABC algorithm against the methods described in Section 5.4 respectively using the LETOR version 2.0 TD2003 and TD2004 datasets. Except for Hybrid ABC, all other results reported in Tables 2 and 3 are obtained from published work and we do not implement those methods here by ourselves. All results reported in Tables 1 and 2 use the officially released LETOR2 datasets and evaluation tools, which enable us to make a direct and a fair comparison.

From Tables 2 and 3 we see that the proposed HybridABC has an efficient performance in terms of MAP scores among the different methods compared. Table 2 shows the results of our proposed algorithm tested with Letor dataset, TREC-TD2003 and table 3 shows the results of our algorithm tested with Letor dataset, TREC-TD2004. Apart from the results of HybridABC all the other results were taken from other researches. The improvements reported by Hybrid ABC over all other methods are statistically significant. Hybrid ABC has a very efficientNDCG@1 and P@1 values in both TD2003 and TD2004 datasets. This implies that the proposed rank learning method (Hybrid ABC) is able to rank relevant documents as the first hit, which is one of the desirable qualities of a ranking function for a search engine. It is interesting to note that on the TD2004 dataset, none of the previously proposed EC-based ranking algorithms (i.e. SwamRank and GPRank) were able to out-perform RankBoost, a non-EC algorithm. However, the proposed rank learning algorithm (HybridABC) goes very close to RankBoost and in some hits outperforms RankBoost. We have represented out results graphically for dataset- TD2003. We have represented the bar representation for the results obtained from dataset TD2003 in Figure 6

using MAP method. In Figure 7 we displayed the results obtained using P@n method for dataset TD2003. In Figure 8 we represented the results obtained by using NDCG@n method for dataset TD2003. For dataset TD2004 we have also prepared graphical representation of our results. Figure 9 shows the results obtained by using MAP method for TD2004 dataset. Figure 10 and Figure 11 shows results for dataset TD2004 using respectively method P@n and NDCG@n. The Figures and Tables representing the results is given below:

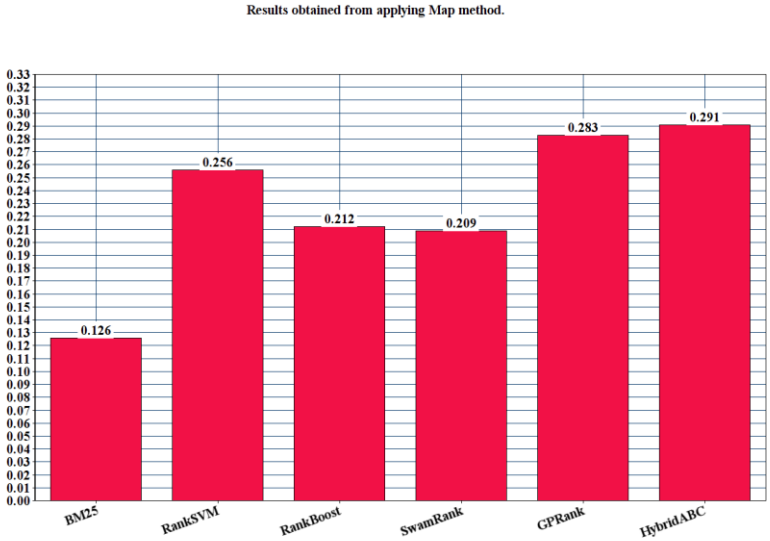

Figure 6: Results obtained from applying MAP method for Dataset TD2003

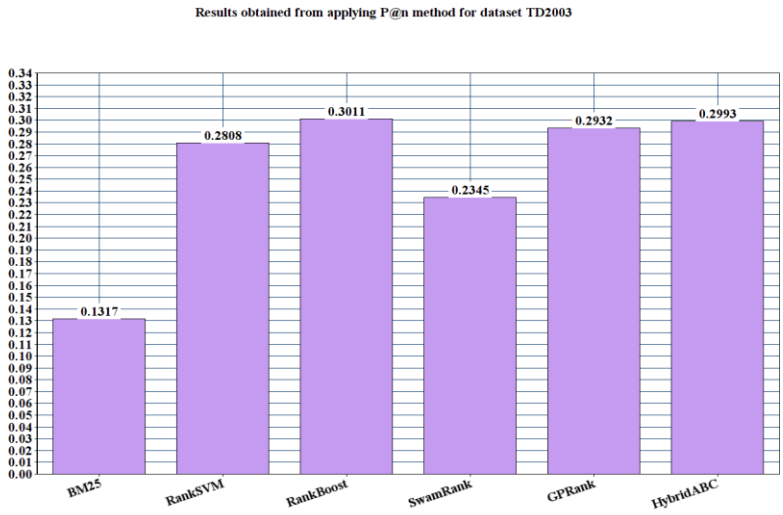

Figure 7: Results obtained from using P@n method for Dataset TD2003

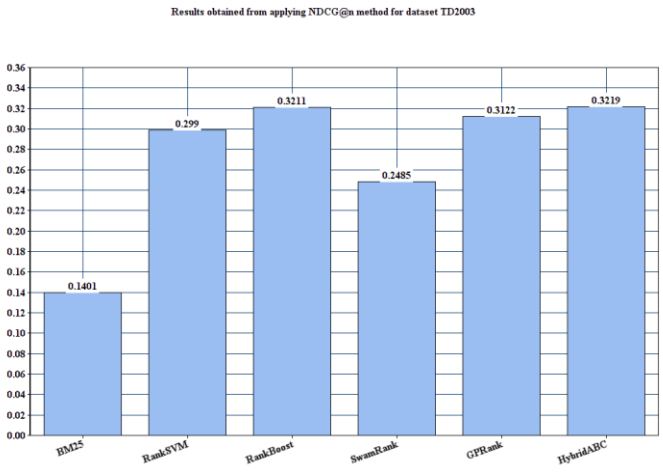

Figure 8: Results for method NDCG@n for Dataset TD2003

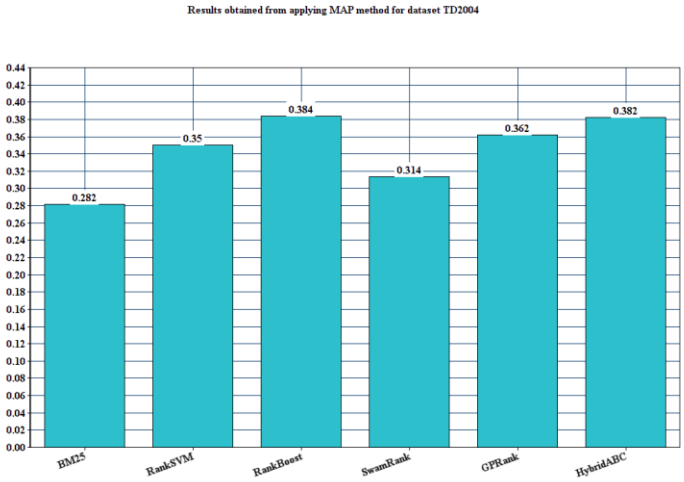

Figure 9 : Results for method MAP for Dataset TD2004

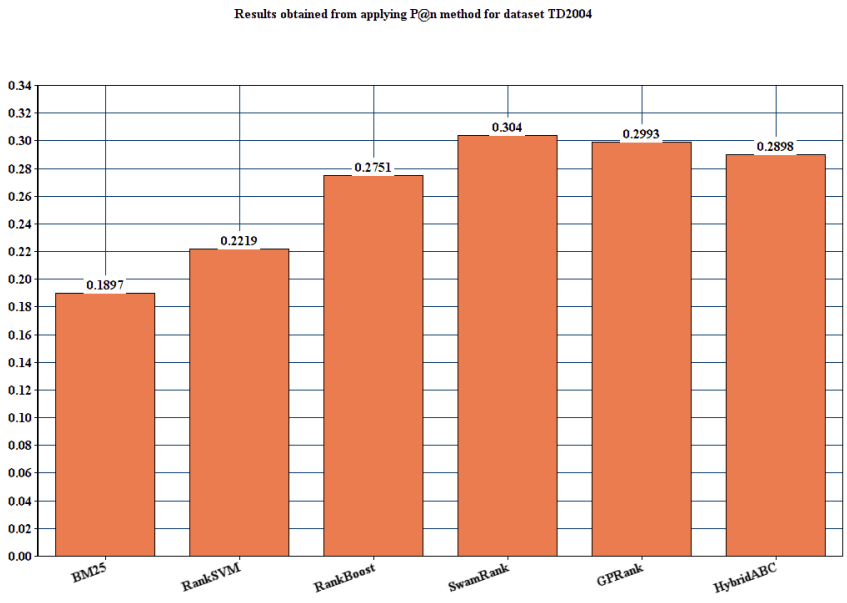

Figure 10: Results obtained using method P@n for dataset TD2004

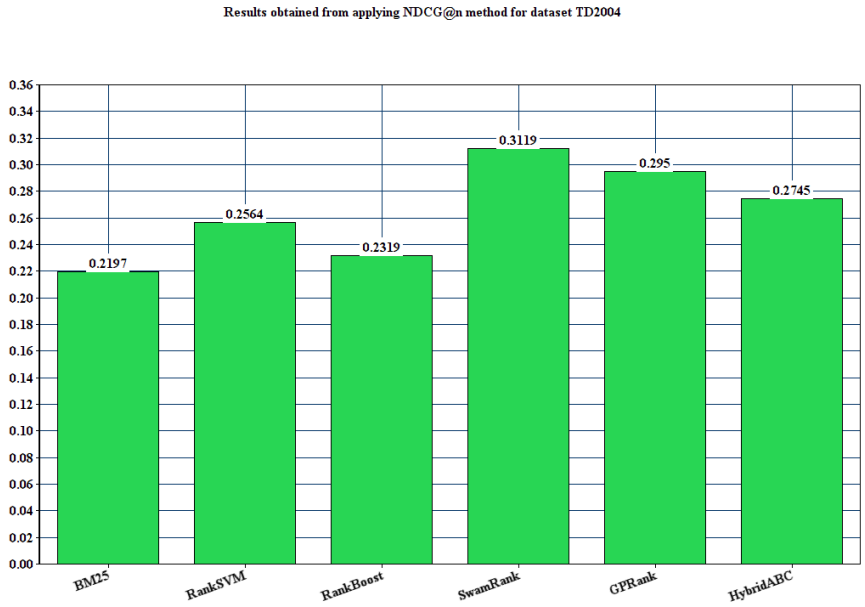

Figure 11: Results obtained using method NDCG@n for dataset TD2004

Method	BM 25	RankSVM	RankBoost	SwamRank	GPRank	HybridABC
Map	0.126	0.256	0.212	0.209	0.283	0.291
p@1	0.120	0.42	0.260	0.453	0.520	0.518
p@2	0.130	0.350	0.270	0.330	0.420	0.400
p@3	0.160	0.340	0.240	0.269	0.370	0.333
p@4	0.145	0.300	0.230	0.223	0.330	0.300
p@5	0.148	0.264	0.220	0.207	0.280	0.280
p@6	0.140	0.243	0.210	0.188	0.270	0.250
p@7	0.129	0.234	0.211	0.185	0.250	0.243
p@8	0.120	0.233	0.193	0.173	0.240	0.237
p@9	0.116	0.218	0.182	0.164	0.230	0.222
p@10	0.109	0.206	0.178	0.151	0.220	0.210
NDCG@1	0.120	0.420	0.260	0.453	0.520	0.540
NDCG@2	0.140	0.370	0.280	0.343	0.450	0.340
NDCG@3	0.176	0.379	0.270	0.307	0.420	0.388
NDCG@4	0.174	0.363	0.272	0.284	0.390	0.356
NDCG@5	0.183	0.347	0.279	0.278	0.380	0.336
NDCG@6	0.184	0.341	0.280	0.271	0.370	0.310
NDCG@7	0.184	0.340	0.287	0.273	0.360	0.300
NDCG@8	0.185	0.345	0.282	0.270	0.350	0.292
NDCG@9	0.186	0.342	0.282	0.267	0.350	0.279
NDCG@10	0.186	0.341	0.285	0.263	0.350	0.267

TABLE 2

Method	BM 25	RankSVM	RankBoost	SwamRank	GPRank	HybridABC
Map	0.282	0.350	0.384	0.314	0.362	0.382
p@1	0.307	0440	0.480	0.400	0.450	0.522
p@2	0.293	0.407	0.447	0.380	0.420	0.500
p@3	0.258	0.351	0.404	0.351	0.380	0.436
p@4	0.243	0.327	0.347	0.317	0.330	0.404
p@5	0.229	0.291	0.323	0.296	0.320	0.385
p@6	0.224	0.273	0.304	0.278	0.300	0.333
p@7	0.210	0.261	0.293	0.253	0.280	0.308
p@8	0.247	0.247	0.277	0.235	0.260	0.308
p@9	0.182	0.236	0.262	0.221	0.250	0.282
p@10	0.175	0.225	0.253	0.215	0.240	0.254
NDCG@1	0.307	0.440	0.480	0.400	0.450	0.692
NDCG@2	0.327	0.433	0.473	0.413	0.440	0.544
NDCG@3	0.314	0.409	0.464	0.404	0.430	0.488
NDCG@4	0.315	0.406	0.439	0.393	0.440	0.458
NDCG@5	0.319	0.939	0.437	0.391	0.440	0.438
NDCG@6	0.325	0.397	0.448	0.394	0.450	0.399
NDCG@7	0.326	0.406	0.457	0.392	0.460	0.377
NDCG@8	0.324	0.410	0.461	0.396	0.470	0.371
NDCG@9	0.332	0.414	0.464	0.397	0.470	0.350
NDCG@10	0.335	0.420	0.472	0.402	0.470	0.328

TABLE 3

6. Feature Analysis

We used Equations in our work. Our proposed method can be interpreted as a weighted linear combination of 44 different features. By inspecting the final weight vector returned by Algorithm 1, we can gain some insight into which features are important for determining the relevance of a document retrieved for a query. The weights learnt by HybridABC using the TD2003 dataset for different features. We then sorted the features in the descending order of their weights.

From the datasets and weights of HostRank 5:9932 ; idf (body)3:938 ; HyperlinkScore(weighted-in-link) 3:798 ; HITS(hub) 3:548 ;tfidf(anchor) 2:571 ; BM25(anchor) 1:870TopicalHITS(authority) 1:727; HyperlinkFeature(weighted-in-link) 1:653 , we see that heuristics that are known to produce better document rankings such as the HostRank, inverse document frequency (idf) of the body text, inbound hyperlinks and HITS are assigned positive efficient weights by HybridABC. The ability of the proposed method to detect salient features for ranking is important if we want to use a large number of features in an information retrieval system. For example, we can prune the trained model based on the weights learnt for different features to improve the speed during test (retrieval) phase, which can be critical for online Web search engines.[2]

7. Conclusion:

In this paper we were successfully able to reflect the findings of our research by showing and comparing our results with numerous other ranking methods like BM25, RankSVM,

RankBoost, SwamRank, GPRank and we used the same Dataset these algorithms's used to evaluate themselves. In the future we plan to update the features of our algorithm to further improve optimization and test out with different parameter settings and more recent versions of Letor datasets, also test it against more contemporary examples of multiobjective evolutionary algorithms.

The purpose of this research was to establish a relatively newer approach to the web-ranking field and one that embraces new concepts for the betterment of web searching, information retrieval, product recommendation etc. From our results obtained from various tests and analysis carried out so far, it has been proved that a hybrid version of the original Artificial Bee Colony algorithm not just stands beside the contemporary ranking methods, but also paves the way for an innovative new approach to web ranking and better performance for the billions of web users today.

Reference

[1]. D. Karaboga, “An idea based on honey bee swarm for numerical optimization,” Techn. Rep. TR06, Erciyes Univ. Press, Erciyes, 2005.

[2]. D. Bollegala, N. Noman, and H. Iba, “RankDE: Learning a Ranking Function for Information retrieval using Differential Evolution”

[3]. D. Karaboga and B. Akay, “A comparative study of artificial bee colony algorithm,”Applied Mathematics and Computation, vol. 214, no. 1, pp. 108–132, 2009.

[4] GAnuradha and Lavanya G Devi. “ Artificial Bee Colony (ABC) Approach for Ranking Web Pages. International Journal of Computer Applications”. 99(1):35-39, August 2014.

[5] G. Zhu and S. Kwong, “Gbest-guided Artificial Bee Colony algorithm for numerical function optimization,” Applied Mathematics and Computation, vol. 217, no. 7, pp. 3166–3173, Dec. 2010.

[6] W. Gao and S. Liu, “A modified Artificial Bee Colony algorithm,” Computers and Operation Research, vol. 39, no. 3, pp. 687–697, Mar. 2012.

[7] International Journal of Machine Learning and Computing, Vol. 5, No. 3, June 2015

[8] Qin, Tao et al. "LETOR: A Benchmark Collection For Research On Learning To Rank For Information Retrieval". Information Retrieval 13.4 (2010): 346-374. Web.

[9] Babu, B.V.; Jehan, M.M.L., "Differential evolution for multi-objective optimization," in Evolutionary Computation, 2003. CEC '03. The 2003 Congress on , vol.4, no., pp.2696-2703 Vol.4, 8-12 Dec. 2003

[10] Deb, K. et al. "A Fast And Elitist Multiobjective Genetic Algorithm: NSGA-II". IEEE Transactions on Evolutionary Computation 6.2 (2002): 182-197. Web.

[11] Nadezda Stanarevic, “Hybridizing artificial bee colony (ABC) algorithm for largescale optimization problems”

[12] Laxmi Choudhary, Bhawani Shankar Burdak. 2008,” Role of Ranking Algorithms for

Information Retrieval ” Discrete Algorithms Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms; Pages: 1010- 1018.

[13] Liu C, Wang G, Xie Q, Zhang Y. Vibration Sensor-Based Bearing Fault Diagnosis Using Ellipsoid-ARTMAP and Differential Evolution Algorithms. *Sensors*. 2014; 14(6):10598-10618

[14] HemaDubey, Prof. B. N. Roy. 2011,”An Improved Page Rank Algorithm based on Optimized Normalization Technique”, International Journal of Computer Science and Information Technologies, Vol. 2 (5) , 2183-2188

[15] Angela Hsiang, Ling-Chen, Yun-Chia Liang, and Jose David Padilla, “An Entropy-Based Upper Bound Methodology for Robust Predictive Multi-Mode RCPSP Schedules”
Entropy 2014, 16(9), 5032-5067.

[16] ManjuPatel1 and ShwetaModi. 2011,” A Survey on Distributed Page Ranking”, International Journal of Chemistry and Applications. ISSN 0974-3111 Volume 3, pp. 201-208