Less-redundant Text Summarization using Ensemble Clustering Algorithm based on GA and PSO

JUNG SONG LEE[1], HAN HEE HAHM[2], SOON CHEOL PARK[1]
[1]Division of Electronics and Information Engineering
Chonbuk National University
[2]Department of Archeology and Cultural Anthropology
Chonbuk National University
567 Baekje-daero, Deokjin-gu Jeonju-si, Jeollabuk-do
REPUBLIC OF KOREA
ei200411147@jbnu.ac.kr, hanheeh @jbnu.ac.kr, scpark@jbnu.ac.kr

Abstract: In this paper, a novel text clustering technique is proposed to summarize text documents. The clustering method, so called 'Ensemble Clustering Method', combines both genetic algorithms (GA) and particle swarm optimization (PSO) efficiently and automatically to get the best clustering results. The summarization with this clustering method is to effectively avoid the redundancy in the summarized document and to show the good summarizing results, extracting the most significant and non-redundant sentence from clustering sentences of a document. We tested this technique with various text documents in the open benchmark datasets, DUC01 and DUC02. To evaluate the performances, we used F-measure and ROUGE. The experimental results show that the performance capability of our method is about 11% to 24% better than other summarization algorithms.

Key-Words: Text Summarization; Extractive Summarization; Ensemble Clustering; Genetic Algorithms; Particle Swarm Optimization

1 Introduction

Generally, automatic summarization techniques can be categorized into extractive and abstractive summarization [21]. An extractive summarization technique is to select the most important sentences from the full text to make short versions [13]. In this technique, the importance of each sentence of a document is decided based on some similarity measures to assign the salience score to the sentence, and the related units with the highest scores are extracted [1]. An abstraction summarization technique usually needs information fusion, and sentence compression and reformulation to paraphrase the contents of the original document [27]. The implementation of abstract summarization techniques requires using heavy machinery from natural language processing for new sentence generation, which is too difficult to get a robust summarization [29]. The quality of an extractive summarization might not be as good as an abstraction summarization, but it is considered well enough for a reader to understand the main ideas of a document, so most of works in this area are based on the extractive summarization.

The extractive summarization techniques use the heuristic rule in order to select the sentences providing the most important information from the document [8]. However, it has a redundancy problem that the selected sentences may have the many same terms due to the high frequencies of those terms. To reduce the redundancy in the summarized sentences, many methods, such CRF [21], Manifold–Ranking [28], NetSum [26], QCS [7] and etc. were proposed.

We applied the clustering technique to reduce redundancy on summarizing results. In this clustering technique, the sentences in a document are clustered according to their similarities into several sentence groups (clusters) and the highest scored sentence of each group is selected as a candidate of the summarized sentences. This technique is very effective to reduce redundancy in the summary. Because one of the candidate sentences includes much of the important content of a cluster (because it is the highest scored sentence in the cluster), and is quite different from the candidate sentences of other clusters (guaranteeing the minimum redundancy).

For sentence clustering, we introduce the ensemble method using both genetic algorithms (GA) and particle swarm optimization (PSO). GA and PSO is well-known for optimization problem but their weakness are the premature convergence. We solve this problem applying different algorithms to the global and the local search algorithms individually. GA is very good for the global search but relatively weak for the local search, and PSO's local search ability is better comparing to GA. Therefore, a well balance between global and local search abilities of two algorithms, GA and PSO, is necessary. We proposed automatic population partitioning (APP) method to solve it. For this, we apply two control parameters (relative distance and distribution coefficient) to regulate the probability of performing local and global searching.

This paper is organized as follows: Details of automatic document summarization based on ensemble method of GA and PSO is described in Section 2. Experiment results are given in Section 3. Conclusions and future works are given in Section 4.

2 Automatic Document Summarization using Sentence Clustering based on Ensemble Method of Automatic Population Partitioning with GA and PSO

First, for our summarization system based on ensemble method of automatic population partitioning (APP) with GA and PSO, sentences are represented by using IR techniques. Second, the sentences are clustered by using APP to reduce the redundancy. Then, the sentences which have the weightiest terms in clusters are selected. Finally, the selected sentences are rearranged in the document for reading.

2.1 Sentence representation and similarity measure between sentences

In most existing document clustering algorithms, documents are represented using the Vector Space Model (VSM) [2], which is formed by the weights of the terms indexed in a document. Equation 1 shows the *n*th document vector whose size is 1 by *t*:

$$d_n = \langle W_{n,1},\ W_{n,2},\ \cdots ,W_{n,t} \rangle, \tag{1}$$

where *t* is the number of the total indexed terms in a corpus and $W_{n,i}$ is the weight of the *i*th term in the *n*th document. Unlike document clustering, extractive summarization technique divides the input document into a set of sentences for the sentence clustering. That is, each document *D* is expressed as sentences sequence:

$$\mathrm{D} = \langle S_1,\ S_2,\ \cdots ,\mathrm{S_n} \rangle, \tag{2}$$

where *n* is number of sentences in *D*. Subsequently, each sentence S_n is represented as:

$$\mathrm{S_n} = \langle \mathrm{TW_{n,1}},\ \mathrm{TW_{n,2}},\ \cdots ,\mathrm{TW_{n,m}} \rangle, \tag{3}$$

where *m* is the number of the total indexed terms in a document and *TW* is the term weight. It shows the *n*th sentence vector whose size is 1 by *m*. *TW* is defined by:

$$\mathrm{TW_{nm}} = \frac{\mathrm{freq}_{\mathrm{\ nm}}}{\mathrm{maxfreq}_{\mathrm{\ nm}}}, \tag{4}$$

where $freq_{ij}$ is term frequency and max$freq_{ij}$ is maximum term frequency in a document. For sentence clustering in VSM, we use cosine measure to compute the similarity between two sentences.

As above, VSM can be applied to sentence representation and similarly. However, it has a drawback. In a sentence, the vector dimension *m* is very large compared to the number of terms. Therefore, the sentence vector has many null components [14]. So, the representation and similarly using VSM is not very efficient for sentence and, we have applied another sentence representation and similarly techniques. Each sentence S_n is defined by:

$$\mathrm{S_n} = \langle \mathrm{T_{n,1}},\ \mathrm{T_{n,2}},\ \cdots ,\mathrm{T_{n,m}} \rangle, \tag{5}$$

where *m* is the number of indexed terms in a sentence S_n. We extracted the indexed terms by using stop words and Porter's stemming. That is, a sentence S_n is represented as sequence of terms existing in the document.

Next, we present a method to measure similarity between sentences using the Normalized Google Distance (NGD) [3]. NGD takes advantage of the number of hits returned by Google search engine to compute the semantic distance between two sentences. NGD is defined the global and local similarity measure between terms in sentences. First, the global similarity measure between terms t_i and t_j is defined by the formula:

$$\mathrm{NGD_g}(\mathrm{t_i},\mathrm{t_j}) = \frac{\max\{\log(\mathrm{f_g}(\mathrm{t_i})), \log(\mathrm{f_g}(\mathrm{t_j}))\} - \log(\mathrm{f_g}(\mathrm{t_i},\mathrm{t_j}))}{\log \mathrm{N_{google}} - \min\{\log(\mathrm{f_g}(\mathrm{t_i})), \log(\mathrm{f_g}(\mathrm{t_j}))\}}, \tag{6}$$

where $f_g(t_i)$ and $f_g(t_j)$ denote for the numbers of web pages containing the search terms t_i and t_j respectively. $f_g(t_i, t_j)$ is the number of web pages containing both terms t_i and t_j. N_{google} is the total number of web pages indexed by Google search engine. Using the definition of global similarity measure between terms as Equation (6), the global sentence similarity measure between sentences S_k and S_l is given by:

$$\mathrm{sim_{global}}(\mathrm{S_k},\mathrm{S_l}) = \frac{\sum_{\mathrm{t_i}\in \mathrm{S_k}} \sum_{\mathrm{t_j}\in \mathrm{S_l}} \mathrm{NGD_{global}}(\mathrm{t_i},\mathrm{t_j})}{\mathrm{m_i m_j}}, \tag{7}$$

where m_i and m_j represent the numbers of terms in sentences S_k and S_l respectively.

Similarly, the local similarity measure between terms t_i and t_j is defined by:

$$\mathrm{NGD_l}(\mathrm{t_i},\mathrm{t_j}) = \frac{\max\{\log(\mathrm{f_l}(\mathrm{t_i})), \log(\mathrm{f_l}(\mathrm{t_j}))\} - \log(\mathrm{f_l}(\mathrm{t_i},\mathrm{t_j}))}{\log \mathrm{N_{google}} - \min\{\log(\mathrm{f_l}(\mathrm{t_i})), \log(\mathrm{f_l}(\mathrm{t_j}))\}}, \tag{8}$$

where $f_l(t_i)$ and $f_l(t_j)$ denote the numbers of sentences containing terms t_i and t_j, respectively, in document D. $f_l(t_i, t_j)$ is the number of sentences containing both t_i and t_j, and n is the total number of sentences in document D. Also, using Equation (8), the local sentence similarity measure is given by:

$$\mathrm{sim_{local}}(\mathrm{S_k},\mathrm{S_l}) = \frac{\sum_{\mathrm{t_i}\in \mathrm{S_k}} \sum_{\mathrm{t_j}\in \mathrm{S_l}} \mathrm{NGD_{local}}(\mathrm{t_i},\mathrm{t_j})}{\mathrm{m_i m_j}}, \tag{9}$$

Finally, the overall sentence similarity measure between sentences S_k and S_l is defined as a product of global and local similarity measures:

$$\mathrm{sim_{NGD}}(\mathrm{S_k},\mathrm{S_l}) = \mathrm{sim_{global}}(\mathrm{S_k},\mathrm{S_l}) \times \mathrm{sim_{local}}(\mathrm{S_k},\mathrm{S_l}). \tag{10}$$

2.2 Generating the proper number of clusters

The number of clusters (topics) in each document is not given before summarization. Thus, we need to determine the proper number of cluster a prior. For this, we used the approach based on the distribution of terms in the sentences which are defined as:

$$\mathrm{k} = \mathrm{n}\frac{|\mathrm{d}|}{\sum_{\mathrm{i}=1}^{\mathrm{n}}|\mathrm{S_i}|} = \mathrm{n}\frac{|\bigcup_{\mathrm{i}=1}^{\mathrm{n}}\mathrm{S_i}|}{\sum_{\mathrm{i}=1}^{\mathrm{n}}|\mathrm{S_i}|}, \tag{11}$$

where $|d|$ is the number of terms in document d and n is number of sentences in d. Authors of the paper [1] provide two cases in which the numbers of clusters are bounded to the k, for clustering n sentences. That is, we always have $1 \le k \le n$. The definition of (11) gives the interpretation of k as the proper number of clusters in terms of average number of terms. Once cluster number is determined by this way, APP is implemented in our study for sentence clustering.

2.3 Ensemble method based on automatic population partitioning with GA and PSO for sentence clustering

Clustering is widely used unsupervised categorization technique partitioning an input space into K regions. One of the mostly used applications of clustering is text document clustering, which categorizes a given large collection of documents into groups of documents having more similar to each other than documents belonging to different groups. It plays a vital role in efficient document organization, summarization, topic extraction, and information retrieval [12].

In the past decade, meta-heuristic algorithms, such as GA, ant colony optimization (ACO), and PSO have been widely used in clustering field. GA is a randomized search and optimization technique that can be used to handle large and complex landscapes guided by the principles of evolution and natural genetics [10]. It can provide near optimal solutions through reproductive evolution of the individual advantage in a population, but its local search ability is relatively weak [25]. Unlike GA, the implementations of optimization and control algorithms based on swarm intelligence e.g., ACO and PSO, have been extensively studied so far. Inspired by the foraging behaviour of ant colonies, ACO targets discrete optimization problems [18]. Shelokar et al. [20] firstly used ACO for clustering, and their experimental results showed that ACO could effectively solve a variety of clustering problems by its good global and local search abilities. However, the searching time is too long. PSO is another efficient swarm intelligence algorithm proposed by Kennedy and Eberhart in 1995 [11]. It simulates the behaviour of bird flocking or fish schooling. In PSO, the potential solution, called particles, moves around in a search space with the velocity updated based on its own experience and the experience of its neighbours (personal best) or the whole swarm (global best) in order to search and determine an optimal solution. In comparing with ACO, PSO is easier to implement and computationally efficient to achieve a fast convergence. Moreover, PSO has a good local search ability compared to GA, but its performance depends highly on the selection of the global best particles [22].

2.4 Clustering by using genetic algorithm

A GA is a robust probabilistic search and optimization technique directed by natural genetics guidelines and evolution principles. It can provide a near optimal solution for objective or fitness function of an optimization problem in a multi-dimensional space.

Text document clustering based on GA can provide appropriate cluster solutions by using the search capability of GA. The performance of GA for text document clustering is better than other clustering algorithms [24]. It is known that a clustering problem can be regarded as an optimization problem that can optimize cluster validity index as an objective or fitness function. It should be noted that a chromosome X_i can be represented as $X_i = (M_{i1}, \ldots, M_{ij}, \ldots, M_{ik})$, where $M_{ij} = (w_1, w_2, \ldots, w_n)$ refers to the centroid vector of the *j*th cluster in the *i*th chromosome, *k* is the number of centroid vectors and, *n* is the total number of terms. Consequently, text document clustering by using GA determines optimal centroid vectors.

In this paper, we use a set type encoding to represent a chromosome and its encoding process is defined by: $X_i = (C_{i1}, \ldots, C_{ij}, \ldots, C_{ik})$, where $\mathrm{C}_{ij} = (S_1,\ S_2, \ldots,\ S_n)$ refers to the sentence group of the *j*th cluster in the *i*th chromosome. *k* is the number of cluster, S_n is a sentence belonging to *j*th cluster, and *n* is the number of sentences in cluster C_{ij}.

An objective or fitness function prescribes the optimality of a solution in GA. That is, the three evolution operators, i.e., selection, crossover, and mutation, of the objective or fitness function determine the evolving direction of the chromosome. In this study, the fitness function for the *i*th chromosome, called the average similarity index, was measured. It was then used to measure the similarity between clusters in group average clustering of one of the hierarchical clustering algorithm [6], and its mean represents the average similarity of all the sentence in each cluster. The average similarity index is defined by:

$$\mathrm{AverageSimilarity(i)} = \frac{\sum_{\mathrm{j}=1}^{\mathrm{k}} \mathrm{ClusterSim}_{\mathrm{j}}}{\sum_{\mathrm{j}=1}^{\mathrm{k}} \mathrm{ClusterSize}_{\mathrm{j}}}, \quad (12)$$

$$\mathrm{ClusterSim_j} = \sum_{\mathrm{d_{jm}} \in \mathrm{C_{ij}}} \sum_{\mathrm{d_{jn}} \neq \mathrm{d_{jm}}} \mathrm{sim}(\mathrm{d_{jm}}, \mathrm{d_{jn}}) \quad , \quad (13)$$

$$ClusterSize_j = \frac{|C_{ij}| \times (|C_{ij}|-1)}{2}, \quad (14)$$

where *ClusterSim*$_j$ and *CluterSize*$_j$ refer to the sum of similarity between sentences in the *j*th cluster and the number of similarity between sentences in the *j*th cluster, respectively. $sim(s_{jm}, s_{jn})$ is the cosine similarity between sentence s_{jm}, s_{jn} and $|C_{ij}|$is the number of sentences in the *j*th cluster in the *i*th chromosome. From Equation (12), we can see that the higher the fitness value, the better the chromosome associated with the solution to the clustering problem.

2.5 Clustering by using particle swarm optimization algorithm

The PSO algorithm, introduced by Kennedy and Eberhart in 1995 [11], is inspired from the concept of the social behavior of a flock of birds. PSO has been proven to be effective for text document clustering [4, 5].

In PSO, the potential solution, called particles, moves around in a multi-dimensional search space with the velocity updated based on its own experience and the experience of its neighbors (personal best) or the whole swarm (global best) in order to search and determine an optimal solution. The velocity and direction of each particle moving along each dimension of the problem space would be altered with the generation of each movement [4]. That is, when a particle moves to a new position, a different candidate solution is generated. Every particle in the swarm is updated using Equations (15) and (16). When applying it to text document clustering, each particle encoded by a cluster centroid vector represents a candidate clustering solution.

In this paper, we use a set type encoding to represent a particle as stated above in the 2.4 and the velocity represented the possibility of sentence moving to a different cluster using Equations (15) and (16).

$$\mathrm{v_{id}(t+1) = wv_{id}(t) + c_1 r_1 (P_{id}(t) - x_{id}(t)) + c_2 r_2 (P_{gd}(t) - x_{id}(t))}, \quad (15)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1), \quad (16)$$

where $v_{id}(t+1)$ is the velocity in the dth dimension of the ith particle for $t + 1$ iteration, w is the inertia weight, c_1 and c_2 as acceleration coefficients are constants, and r_1 and r_2 are two random numbers in the interval [0, 1] applied to the ith particle. P_{id} is the personal best with respect to the ith particle, P_{gd} is the global best in whole swarm, and x_{id} represents the ith particle for $t + 1$ iteration.

2.6 Automatic Population Partitioning

In data clustering, almost all stochastic optimization algorithms, such as GA, ACO and PSO have suffered from low accuracy and premature convergence. That is because, in the search space, each of these algorithms has different optimization ability, and the balance between global and local search abilities is critical to the success of an optimization problem [23]. Thus, in order to achieve global optimization, both global and local search abilities are required: the combination of different meta-heuristic algorithms for improving optimization ability is a hot spot in the research field of clustering.

In this paper, we propose an ensemble method combining GA and PSO to deal with the clustering problem. GA has global search ability but its local search ability is relatively weak and PSO's local search ability is better in comparison with GA. Therefore, a well balance between global and local search abilities is obtained through automatic population partitioning (APP) proposed this study.

Generally, in a population, an individual with high fitness value is considered as good global optimum candidates. Therefore, a local optimum searching approach, such as PSO, is applied to this individual. On the contrary, individual having less likelihood of achieving good fitness evaluation is considered as poor global optimum candidates. Here, GA is applied to this individual to improve the algorithm capacity for global exploration.

For the APP proposed in this paper, we apply two control parameters, i.e., relative distance G and distribution coefficient Var, to regulate the probability of performing local and global searches of each individual X_i of the population in the current generation i. Specifically, by using these control parameter, APP automatically determines the global or local searching of each individual and applies different techniques used in optimization to each of them, such as GA and PSO.

The relative distance between the fitness value of the individual X_i and the best fitness value in the current generation is defined by:

$$\mathrm{G = (fit_{max} - fit(X_i)) / (fit_{max} - fit_{min})}, \quad (17)$$

where fit_{max} and fit_{min} represent the maximum and minimum values of fitness, respectively.

From its definition, we divide the value of G into two intervals:

If $fit(X_i)$ is close to fit_{max}, and G is small, the individual X_i will perform a local search by using PSO.

If $fit(X_i)$ is far from fit_{max}, and G is large, the individual X_i will perform a global search by using GA.

Var is defined to depict the distribution of a population in the current generation, which is used to detect whether the population is converging to an optimum:

$$\mathrm{Var = (fit_{max} - fit_{avg})/(fit_{max} - fit_{min})}, \quad (18)$$

where fit_{max}, $fit_{avg,}$ and fit_{min} represent the maximum, average, and minimum values of fitness, respectively. It is intuitive that when fit_{avg} is close to fit_{max} and distant from fit_{min}, the distribution of the population will be high and may cause premature convergence; thus, we need to expand global search, and vice versa.

Using G and Var, the probability of X_i to perform a local search by using PSO ($P_{pso}(X_i)$) and the probability of X_i to perform a global search by using GA ($P_{ga}(X_i)$) are defined by:

$$\mathrm{P_{pso}(X_i) = k_1 \cdot \frac{Var}{G} = k_1 \cdot \frac{(fit_{max} - fit_{avg})}{(fit_{max} - fit(X_i))}}, \quad (19)$$

$$\mathrm{P_{GA}(X_i) = k_2 \cdot \frac{G}{Var} = k_2 \cdot \frac{(fit_{max} - fit(X_i))}{(fit_{max} - fit_{avg})}}, (20)$$

where k_1 and k_2 are real constants between 0 and 1. Thus, we have:

if $fit(X_i) \geq fit_{avg}$, X_i is good individual, **then** $P_{pso}(X_i) = k_1 \cdot (fit_{max} - fit_{avg}) / (fit_{max} - fit(X_i))$ and $P_{ga}(X_i) = 0$;

else if $fit(X_i) < fit_{avg}$, X_i is bad individual, **then** $P_{pso}(X_i) = 0$ and $P_{ga}(X_i) = k_2 \cdot (fit_{max} - fit(X_i)) / (fit_{max} - fit_{avg})$.

The steps of the APP algorithm can be summarized as:

Step 1: Setting parameters: proper number of clusters *k* using Equation (11), parameter *p* denotes the population size, and parameter *i* denotes the maximum iterations.

Step 2: Initializing population: Create a population P.

Step 3: (a) Calculate the fitness value of each individual of population in generation *i* based on equation (12).
(b) Divide the population by using APP: Determine the probability of each individual in order to perform global search by using GA and local search by using PSO based on equation (19) and (20), respectively.
(c) Assign individuals of the global search part in the population to the chromosomes of GA and perform GA until one iteration.
(d) Assign individuals of the local search part in the population to particles of PSO and perform PSO until one iteration.
(e) Generate offspring population by combining both GA and PSO output individuals.

Step 4: Repeat step 3 until termination conditions are satisfied.

Step 5: Output the final solution obtained by the best individual in the last generation.

2.7 Sentence selection and rearrangement from clusters of sentences for reading

To select the important sentence in a sentence cluster, we use the weights of sentences in each cluster proposed in the paper of Pavan and Pelillo [19]. The Weight of Sentence S_i in sentence cluster C_p will be defined by the following recursive formula as:

$$\mathrm{WOS}_{\mathrm{C_p}}(\mathrm{S_i}) = \begin{cases} 1, & \text{if } |\mathrm{C_p}| = 1 \\ \sum_{\mathrm{S_j} \in \mathrm{C_p}} \Phi_{\mathrm{C_p}}(\mathrm{S_j}, \mathrm{S_i})\, \mathrm{W}_{\mathrm{C_p}}(\mathrm{S_j}), & \text{otherwise} \end{cases} \quad . \qquad (21)$$

where, C_p is nonempty sentence cluster and S_i, S_j are sentences in C_p.

Subsequently, $\Phi_{Cp}(S_j, S_i)$ is:

$$\Phi_{\mathrm{C_p}}(\mathrm{S_j}, \mathrm{S_i}) = \mathrm{sim_{NGD}}(\mathrm{S_j}, \mathrm{S_i}) - \mathrm{awdeg}_{\mathrm{C_p}}(\mathrm{S_j}) \quad . \qquad (22)$$

And $awdeg_{Cp}(S_j)$ is:

$$\mathrm{awdeg}_{\mathrm{C_p}}(\mathrm{S_j}) = \frac{1}{|\mathrm{C_p}|} \sum_{\mathrm{S_i} \in \mathrm{C_p}} \mathrm{sim_{NGD}}(\mathrm{S_j}, \mathrm{S_i}) \quad . \qquad (23)$$

Consequently, top ranked sentences are selected in sentence cluster reversed order of WOS_{Cp} value.

The summary is provided by compounding the important sentences extracted from each sentence cluster. But, it is needed to rearrange the sentences for reading. Each sentence cluster has the information of the indices of the sentences which are the same as the sequence order as in a document. After selecting the weightiest sentences in the clusters, we sort the sentences with their indices and then return the sentences in the sorted order.

3 Experiment Results

3.1 Datasets

We conduct our method of APP for extractive summarization on two document datasets DUC01 and DUC02 and the corresponding 100-word summaries generated for each document. The DUC01 and DUC02 as the most-widely adopted benchmark datasets in the document summarization are the open source datasets published by Document Understanding Conference (http://duc.nist.gov). The DUC01 and DUC02 contain 147 and 567 documents-summary pairs respectively. These datasets are clustered into 30 and 59 topics, respectively. In those document datasets, stop word removal and the terms were stemmed using Porter's stemming.

3.2 Evaluation metrics

To evaluate the performances of the algorithms, we use two measurements. The first measurement is *F-measure* [9] which uses a generic metric to evaluate the performance of IR. The summarization precision $P_{summary}$, recall $R_{summary}$ are defined as:

$$\mathrm{P_{summary}} = \frac{|\mathrm{Sums_{reference}} \cap \mathrm{Sums_{candidate}}|}{|\mathrm{Sums_{candidate}}|}, \qquad (24)$$

$$\mathrm{R_{summary}} = \frac{|\mathrm{Sums_{reference}} \cap \mathrm{Sums_{candidate}}|}{|\mathrm{Sums_{reference}}|} \quad , \qquad (25)$$

where $|Sums_{candidate}|$ stands for the number of the candidate summaries, and $|Sums_{reference}|$ stands for the number of the reference summaries. $|Sums_{reference} \cap Sums_{candidate}|$ is the number of the matching pairs between candidate summaries and reference summaries. *F-measure*$_{summary}$ is subsequently given by:

$$\mathrm{F-measure}_{\mathrm{summary}} = \frac{2\mathrm{P}_{\mathrm{summary}} \times \mathrm{R}_{\mathrm{summary}}}{\mathrm{P}_{\mathrm{summary}} + \mathrm{R}_{\mathrm{summary}}}, \quad (26)$$

The second measurement is the *ROUGE* toolkit [15]. It has been shown that *ROUGE* is very effective for measuring document summarization and it measures the summary quality too by counting the overlapping units between reference summary and candidate summary. *ROUGE-N* measure is given by:

$$\mathrm{ROUGE-N} = \frac{\sum_{S \in \mathrm{Summary}_{\mathrm{refer}}} \sum_{\mathrm{N-gram} \in S} \mathrm{Count}_{\mathrm{match}}(\mathrm{N-gram})}{\sum_{S \in \mathrm{Summary}_{\mathrm{refer}}} \sum_{\mathrm{N-gram} \in S} \mathrm{Count}(\mathrm{N-gram})}, \quad (27)$$

where *Count*(*N-gram*) is the number of *N*-grams in reference summaries, and *Count*$_{match}$(*N*-gram) is the maximum number of *N*-grams co-occurrence between reference summary and candidate summary. *ROUGE-N* compares *N*-grams between these two summaries, and counts the number of matches. *N* stands for the length of *N-gram*. In our experiment, we use two the *ROUGE* metrics, *N* is set as 1 and 2, that is, unigram metric *ROUGE*-1 and bigram metric *ROUGE*-2 are applied.

3.3 Performance and discussion

In this section, we compare the summary performances of APP with those of other five methods, such as CRF [21], Manifold–Ranking [28], NetSum [26], QCS [7], and SVM [30] which are widely used in the automatic document summarization.

Table 1. Summarization performance on DUC01.

Methods	F-measure	ROUGE-1	ROUGE-2
APP	**0.50213**	**0.50021**	**0.20034**
CRF	0.46405	0.45525	0.17665
Manifold Ranking	0.43365	0.42865	0.16354
NetSum	0.47014	0.46231	0.16698
QCS	0.44192	0.43852	0.18457
SVM	0.44628	0.43254	0.17002

Table 2. Summarization performance on DUC02.

Methods	F-measure	ROUGE-1	ROUGE-2
APP	**0.49882**	**0.48673**	**0.14228**
CRF	0.46003	0.44401	0.10873
Manifold Ranking	0.41926	0.42536	0.10528
NetSum	0.46158	0.45562	0.11254
QCS	0.42116	0.45002	0.10547
SVM	0.43152	0.43785	0.10745

Table 1 and Table 2 show the results of all the methods in terms *ROUGE*-1, *ROUGE*-2, and *F-measure* metrics on DUC01 and DUC02 datasets, respectively. From Table 1 and Table 2, we can see that the performances of APP are better than those of other five methods in terms of *F-measure, ROUGE*-1 and *ROUGE*-2.

We also compare APP with other five methods in Table 3. In order to show the improvements of APP with other five methods, we use relative improvement as: $\frac{our\ method - other\ methods}{\text{other methods}} \times 100$. The positive sign (+) stands for improvement, and the negative sign (-) stands for the opposite. Specifically, the performance of APP is around 12% better in terms of the *F-measure*, around 11% better in terms of the *ROUGE*-1 and around 24% better in terms of the *ROUGE*-2 than other algorithms.

Table 3. Comparison of Summarization performance.

Datasets	**Metrics**	**CRF**	**Manifold Ranking**	**NetSum**	**QCS**	**SVM**
DUC01	F-measure	(+)8.21%	(+)15.79%	(+)6.8%	(+)13.62%	(+)12.51%
	ROUGE-1	(+)9.88%	(+)16.69%	(+)8.2%	(+)14.07%	(+)15.64%
	ROUGE-2	(+)13.41%	(+)22.5%	(+)19.98%	(+)8.54%	(+)17.83%
DUC02	F-measure	(+)8.43%	(+)18.98%	(+)8.07%	(+)18.44%	(+)15.6%
	ROUGE-1	(+)9.62%	(+)14.43%	(+)6.83%	(+)8.16%	(+)11.16%
	ROUGE-2	(+)30.86%	(+)35.14%	(+)26.43%	(+)34.90%	(+)32.42%

Table 4. Summarization Result of APP using NGD, Cosine and Euclidean measures.

Datasets	**Measures**	**F-measure**	**ROUGE-1**	**ROUGE-2**

DUC01	NGD	0.50213	0.50021	0.20034
	Cosine	0.49254	0.48722	0.19267
	Euclidean	0.46228	0.45714	0.17228
	Improvement (Cosine)	(+)1.95%	(+)2.67%	(+)3.98%
	Improvement (Euclidean)	(+)8.62%	(+)9.42%	(+)16.29%
DUC02	NGD	0.49882	0.48673	0.14228
	Cosine	0.48439	0.47539	0.13529
	Euclidean	0.46211	0.45558	0.11285
	Improvement (Cosine)	(+)2.98%	(+)2.39%	(+)5.17%
	Improvement (Euclidean)	(+)7.94%	(+)6.84%	(+)26.08%

In Table 4, we compare the performances of APP using different similarity measures (Cosine, Euclidean, and NGD) to test the effectiveness of the NGD-based dissimilarity measure. Consequently, APP with NGD performs better than Cosine and Euclidean measures.

4 Conclusions

In this paper, an ensemble method based on APP for sentence clustering is used in order to improve the performance of summarization. Almost all stochastic optimization algorithms for solving clustering problems suffer from low accuracy and premature convergence. GA has a global search ability to determine a global optimal solution, but its local search ability is relatively weak. On the contrary, PSO's local search ability is better when compared to GA. Thus, we take advantage of the search abilities of both these algorithms and combine GA and PSO to overcome the premature convergence problem. This App is compared to several existing summarization methods on the open DUC01 and DUC01 datasets. Since the conventional document similarity measures are not suitable for computing similarity between sentences, a normalized Google distance is used. We tested them with various methods (five summarization methods) and various datasets (DUC01 containing 147 and DUC02 containing 567) to prove their performances further. Consequently, APP showed higher summarization performances than other methods.

References:

[1] Aliguliyev, R. M., A new sentence similarity measure and sentence based extractive technique for automatic summarization, *Expert System with Applications*, Vol.36, No.4, 2009, pp. 7764-7772.

[2] Choi, L. C., Choi, K. Ung., and Park, S. C., An automatic semantic term-network construction system, *In International Symposium on Computer Science and its Applications*, 2008, pp. 48-51.

[3] Cilibrasi, R. L. and Vitányi, P. M., The Google similarity distance, *IEEE Transactions on Knowledge and Data Engineering*, Vol.19, No.3, 2007, pp. 370-383.

[4] Cui, X. and Potok, T. E., Document clustering analysis based on hybrid PSO+ K-means algorithm, *Journal of Computer Sciences*, 2007, pp. 27-33.

[5] Cui, X., Potok, T. E., and Palathingal, P., Document clustering using particle swarm optimization, *In Proceedings 2005 IEEE Swarm Intelligence Symposium*, 2005, pp. 185-191.

[6] Cutting, D. R., Karger, D. R., Redersen, J. O., and Tukey, J. W., Scatter/gather: A cluster-based approach to browsing large document collections, *In Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1992, pp. 318-329

[7] Dunlavy, D. M., O'Leary, D. P., Conroy, J. M., and Schlesinger, J. D., QCS: A system for querying, clustering and summarizing documents, *Information Processing and Management*, Vol.43, No.6, 2007, pp. 1588-1605.

[8] Fattah, M. A. and Ren, F., GA, MR, FFNN, PNN and GMM based models for automatic text summarization, *Computer Speech Language*, Vol.23, No.1, 2009, pp. 126-144.

[9] Fragoudis, D., Meretakis, D., and Likothanassis, S., Best terms: an efficient feature-selection algorithm for text categorization, *Knowledge Information System*, Vol.8, No.1, 2005, pp. 16-33.

[10] Holland, J. H., Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence, University of Michigan Press, 1975.

[11] James K. and Russell E., Particle swarm optimization, *In Proceedings of IEEE*

International Conference on Neural Networks, 1995, pp. 1942–1948.

[12] Kowalski, G., Information retrieval systems: Theory and implementation, *Computer Mathematics Applications*, Vol.5, No.35, 1998.

[13] Kupiec, J., Pedersen, J., and Chen, F., A trainable document summarizer, *In Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1995, pp. 68-73.

[14] Li, Y.,Luo, C., and Chung, S. M., Text clustering with feature selection by using statistical data, *IEEE Transactions on Knowledge and Data Engineering*, Vol.20, No.5, 2008, pp. 641-652.

[15] Lin, C. Y. and Hovy, E., Automatic evaluation of summaries using N-gram co-occurrence statistics, *In Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, 2003, pp. 71-78.

[16] Mihalcea, R. and Ceylan, H., Explorations in automatic book summarization, *In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2007, pp. 28-30.

[17] Mitra, V., Wang, C. J., and Banerjee, S., Text classification: A least square support vector machine approach, *Applied Soft Computing*, Vol.7, No.3, 2007, pp. 908-914.

[18] Mohan, B. C. and Baskaran. R., A survey: Ant colony optimization based recent research and implementation on several engineering domain, *Expert System with Applications*, Vol.39, No.4, 2012, pp. 4618-4627.

[19] Pavan, M. and Pelillo, M., Dominant sets and pairwise clustering, *IEEE Transactions on Pattern Analysis*, Vol.29, No.1, 2007, pp. 167-172.

[20] Shelokar, P.S., Jayaraman, V.K., and Kulkarni, B.D., An ant colony approach for clustering, *Analytica Chimica Acta*, Vol.509, No.2, 2004, pp. 187-195.

[21] Shen, D., Sun, J. T., Li, H., Yang, Q., and Chen, Z., Document summarization using conditional random fields, *In Proceedings of IJCAI*, 2007, pp. 2862-2867.

[22] Shi, Y. and Eberhart, R., A modified particle swarm optimizer, *In Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence, the 1998 IEEE International Conference on*, pp. 69-73.

[23] Shi, Y. and Eberhart, R., Fuzzy adaptive particle swarm optimization, *In Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, pp. 101-106.

[24] Song, W. and Park, S. C., Genetic algorithm for text clustering based on latent semantic indexing, *Computer and Mathematics with Applications*, Vol.57, No.11, 2009, pp. 1901-1907.

[25] Song, W., Qiao, Y., Park, S. C., and Qian, X., A hybrid evolutionary computation approach with its application for optimizing text document clustering, *Expert System with Applications*, Vol.42, No.5, 2015, pp. 2517-2524.

[26] Svore, K. M., Vanderwende, L., and Burges, C. J., Enhancing single-document summarization by combining RankNet and third-party sources, *In Proceedings of the EMNLP-CoNLL*, 2007, pp. 448-457.

[27] Wan, X., Using only cross-document relationships for both generic and topic-focused multi-document summarizations, *Information Retrieval*, Vol.11, No.1, 2008, pp. 25-49.

[28] Wan, X., Yang, J., and Xiao, J., Manifold-ranking based topic-focused multi-document summarization, *In Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2007, pp. 2903-2908.

[29] Yang, S. and Park, S. C., Generation of Non-redundant Summary Based on Sum of Similarity and Semantic Analysis, *In Information Retrieval Workshop*, 2005, pp. 11-15

[30] Yeh, J. Y., Ke, H. R., Yang, W. P., and Meng, I. H., Text summarization using a trainable sum-marizer and latent semantic analysis, *Information Processing Management*, Vol.41, No.1, 2005, pp. 75-95.