Hindawi Publishing Corporation
Applied Computational Intelligence and Soft Computing
Volume 2011, Article ID 351498, 13 pages
doi:10.1155/2011/351498

Research Article

pSum-SaDE: A Modified *p*-Median Problem and Self-Adaptive Differential Evolution Algorithm for Text Summarization

Rasim M. Alguliev, Ramiz M. Aliguliyev, and Chingiz A. Mehdiyev

Institute of Information Technology of Azerbaijan National Academy of Sciences, B. Vahabzade Street, 9, AZ1141 Baku, Azerbaijan

Correspondence should be addressed to Ramiz M. Aliguliyev, r.aliguliyev@gmail.com

Received 11 May 2011; Revised 26 July 2011; Accepted 27 August 2011

Academic Editor: Chuan-Kang Ting

Extractive multidocument summarization is modeled as a modified *p*-median problem. The problem is formulated with taking into account four basic requirements, namely, relevance, information coverage, diversity, and length limit that should satisfy summaries. To solve the optimization problem a self-adaptive differential evolution algorithm is created. Differential evolution has been proven to be an efficient and robust algorithm for many real optimization problems. However, it still may converge toward local optimum solutions, need to manually adjust the parameters, and finding the best values for the control parameters is a consuming task. In the paper is proposed a self-adaptive scaling factor in original DE to increase the exploration and exploitation ability. This paper has found that self-adaptive differential evolution can efficiently find the best solution in comparison with the canonical differential evolution. We implemented our model on multi-document summarization task. Experiments have shown that the proposed model is competitive on the DUC2006 dataset.

1. Introduction

Automatic document summarization has drawn increasing attention in the past with rapid growth of the Internet and electronic government. The explosion of electronic documents has led to information overload, implying that finding and using information efficiently and effectively has become a pressingly practical problem. The information overload can be reduced by text summarization together with conventional search engines to efficiently access the relevance of retrieved documents. Automatic document summarization aims to condense the original text into its essential content and to assist in filtering and selection of necessary information [1].

Depending on the number of documents to be summarized, the summary can be a single document or a multidocument. Single-document summarization can only condense one document into a shorter representation, whereas multidocument summarization can condense a set of documents into a summary. Multidocument summarization can be considered as an extension of single-document summarization and used for precisely describing the information contained in a cluster of documents and facilitate users to understand the document cluster. Since it combines and integrates the information across documents, it performs knowledge synthesis and knowledge discovery and can be used for knowledge acquisition [2].

There are two approaches for document summarization: supervised and unsupervised [3]. The supervised approaches treat document summarization as a classification and the task formalize as identifying whether a sentence should be included in the summary or not. However, they require training samples. The unsupervised methods usually utilize clustering algorithms to score the sentences in the documents by combining a set of predefined features [4–6].

The summarization task can also be categorized as either generic or query oriented. A query-oriented summary presents the information that is more relevant to the given queries, while a generic summary gives an overall sense of the document's content [7, 8].

In this paper, we focus on the unsupervised generic text summarization, which generates a summary by extracting salient sentences in given document(s). We represent generic text summarization as a modified *p*-median problem. One of the advantages of this approach is that it directly discovers key sentences in the given collection and covers the main content of the original source(s). Another advantage of our model is that it can reduce redundancy in the summary. In this paper, a self-adaptive differential evolution (DE) algorithm is created to solve the optimization problem. The performance of the proposed approach is tested on the DUC2006 dataset and is compared with baseline systems. The effectiveness of the proposed approach is demonstrated.

The rest of this paper is organized as follows. Section 2 gives brief review of related work. Formulation of sentence selection problem for text summarization is introduced in Section 3. Section 4 describes a modified DE algorithm to solve the optimization problem. The numerical experiments and results are given in Section 5. Finally, we conclude our paper in Section 6.

2. Related Work

Generally, document summarization methods can be divided into two categories: abstractive and extractive [4, 9]. Extractive summarization is a simple but robust method for text summarization and it involves assigning saliency scores to some textual units of the documents and extracting those with highest scores. Abstraction can be described as reading and understanding the text to recognize its content, which is then compiled in a concise text. In general, an abstract can be described as summary comprising concepts/ideas taken from the source that are then reinterpreted and presented in a different form, whilst an extract is a summary consisting of units of text taken from the source and presented verbatim [10].

Many approaches have been proposed for document summarization based on the diversity. The pioneer work for diversity-based document summarization is MMR (Maximal Marginal Relevance); it was introduced by Carbonell and Goldstein [11]. The method MMR summarizes document by calculating the cosine similarity between a given query and a document and the cosine similarity between the currently selective sentence and the previously selected sentence. MMI (Maximal Marginal Importance) [12] is also a diversity-based text summarization method for summary generation. It depends on the extraction of the most important sentences from the original text. Most features (i.e., sentence centrality; title feature; word sentence score; keyword feature; similarity to first sentence) used in this method are combined in a linear combination to show the importance of the sentence. CollabSum [13] reduces redundancy by discarding the highly overlapping sentences with already extracted highly ranked sentences.

Clustering has become an increasingly important topic with explosion of information available via the Internet and electronic government services. It is an important tool in text mining and knowledge discovery. Its ability to automatically group similar textual objects together enables one to discover hidden similarity and key concepts, as well as to summarize a large amount of text into a small number of groups [14]. In [15], the clustering is used as an effective tool for finding the diversity among the sentences. This work firstly clusters the sentences and uses the obtained sentence clusters to generate a summary. The paper [16] proposes a query-based multidocument summarization method, using NNM semantic features and the clustering method. The works [17–19] use the NGD-based sentence similarity measure to cluster sentences, so that related sentences can be joined together in a briefer representation of the original text. Recently, a new language model, factorization with given bases (FGB) [20], is proposed for document clustering and summarization by making use of both word-document matrix and word-sentence matrix.

Ouyang et al. [21] apply a machine learning approach to topic-based summarization by regarding sentence scoring as a regression problem. The regression function is learned from the Support Vector Regression (SVR) model, which is the regression type of Support Vector Machine (SVM) and is capable of building state-of-art optimum approximation functions. It provides a way of combining the features automatically and effectively. To save the costly manual annotation time and effort, they construct training data automatically from the document sets where the reference summaries generated by human have been provided. The paper [22] investigates the problem of multi-topic-based query-oriented summarization. Authors of this work formalize the major tasks and propose a probabilistic approach to solve the tasks. They study two strategies for simultaneously modeling document contents and the query information and present four methods to score sentences in the documents based on the learned topic models. HierSum [23] is a generative summarization method based on topic models, which uses sentences as an additional level. Using an approximation for inference, sentences are greedily added to a summary so long as they decrease Kullback-Leibler divergence. Ranking plays an important role in information retrieval and natural language processing applications. The main contributions of the work [24] are threefold: (1) presents a "rank-learn-combine" unsupervised ensemble-ranking framework, namely, interactive ranking (iRANK); (2) explores two ranking refinement strategies that either utilize the feedback as an additional ranking feature or to ensure rank consistency during refinement; (3) proposes two new sentence-ranking algorithms based on iRANK for query-focused summarization. An approach, proposed in [25], for producing a summary consists of three steps. First, it associates sentences and queries with a representation in the latent topic space of a PLSA model by estimating their mixing proportions. It then computes several sentence-level features based on the similarity of sentence and query distributions over latent topics. Finally, it combines individual feature scores linearly into an overall sentence score to create a ranking, which we use to select sentences for the summary. Hybrid hierarchical summarizer, HybHSum [26], is based on a hybrid learning approach to extract sentences for generating summary. It discovers hidden topic distributions of sentences in a given document cluster along with provided summary sentences based on hierarchical Latent Dirichlet Allocation (LDA), which is a generalization of LDA. Contributions of this work are as follows: (1) construction of hierarchical probabilistic model designed to discover the topic structures of all sentences;

(2) representation of sentences by metafeatures to characterize their candidacy for inclusion in summary text; (3) implementation of a feasible inference method based on a regression model to enable scoring of sentences in test document clusters without retraining.

With the publishing of work [27], an optimization approach began to be applied actively in extractive document summarization. It is directly connected with character of the extractive summarization; in other words, identification of informative sentences in documents by the nature is an optimization problem. Takamura and Okumura [28] represented text summarization as maximum coverage problem with knapsack constraint. McDonald [29] formalized text summarization as a knapsack problem and obtained the global solution and its approximate solutions. Wang et al. [30] proposed a Bayesian sentence-based topic model (BSTM) for multidocument summarization by making use of both the word-document and word-sentence associations. The BSTM models the probability distributions of selecting sentences given topics and provides a principled way for the summarization task. Tao et al. [31] have designed word-based and sentence-based association networks and proposed word and sentence weighting approaches based on how much cooccurrence information they contain and applied to text summarization. In [32], text summarization formalized as a budgeted median problem. This model covers the whole document cluster through sentence assignment, since in this model one of the selected sentences as much as possible represents every sentence. An advantage of this method is that it can incorporate asymmetric relations between sentences in a natural manner. Huang et al. [33] consider document summarization as a multiobjective optimization problem. In particular, they formulate four objective functions, namely, content coverage, relevancy, redundancy, and text coherence. They measure the possible summaries based on the identified core terms and main topics (i.e., a cluster of semantically or statistically related core terms).

3. Problem Statement and Its Mathematical Formulation

3.1. Problem Statement. In this section, we present our approach towards all of the four aspects of summarization as follows.

(1) *Relevancy*. A good summary should contain the most important information, that is, selected sentences should be relevant to the main content of the source.

(2) *Content Coverage*. A summary should contain every important aspect of the document. By considering coverage, the information loss in summarization can be minimized.

(3) *Diversity*. A good summary should be concise and contain as few redundant sentences as possible, that is, two sentences providing similar information should not be both present in the summary. In practice, enforcing diversity in summarization can effectively reduce redundancy among the sentences.

(4) *Length*. A summary should be bounded in length.

Optimizing all four properties jointly is a challenging task and is an example of a global summarization problem. That is why the inclusion of relevant sentences relies not only on properties of the sentences themselves, but also on properties of every other sentence in the summary [33]. Our goal is to choose from a set of documents a subset of sentences so that the created summary has satisfied the above four aspects. In our study, this goal has been reached with modifying of the p-median problem.

To apply a p-median problem to sentence-extraction-based document summarization, each sentence in a document collection should be presented as a point in Euclidean space and defined a measure to calculate a similarity between points (sentences).

3.2. Sentence Representation and Similarity Measure. Let a document collection be decomposed into a set of sentences $D = \{s_1, s_2, \ldots, s_n\}$, where n is the number of sentences, s_i denotes ith sentence in D. For calculation of similarity between textual units, each of them should be presented as a vector. The vector space model is the most known representation scheme for textual units. The vector space model represents textual units by counting terms or sequence of terms. Let $T = \{t_1, t_2, \ldots, t_m\}$ represent all the distinct terms occurring in the collection, where m is the number of different terms.

Vector Space Model. The standard vector space model (hereinafter referred to as VSM) is a model for representing text in a vector space based on the bag of words approach. It was first presented as a model for Information Retrieval (IR) in [34]. In VSM, text units of a corpus are represented by vectors. Traditionally a whole document is used as a text unit, but any other text unit like paragraphs or sentences can be used just as well. Each dimension of a vector corresponds to a term that is present in the corpus. A term might be, for example, a single word, N-gram, or a phrase. If a term occurs in a sentence, the value of that dimension is nonzero. Values can be binary ($1 \rightarrow$ term is present in the sentence, $0 \rightarrow$ term is not present in the sentence), frequencies of terms in the sentence, or term weights [35].

Term Weighting. The idea behind term weighting is to assign a weight to represent the importance of a term. The raw frequency of a term only states how often a term occurs in a document without measuring the importance of that term within the sentence or within the whole collection. Different weighting schemes are available. The most common and popular one is the tf-isf weighting scheme. It combines local and global weighting of a term.

*Local Term Weighting (*tf*).* It measures the importance of a term within a sentence:

$$\mathrm{tf}_{ik} = \mathrm{freq}_{ik}, \tag{1}$$

where freq_{ik} is the frequency of term t_k in sentence s_i. This formula assigns a higher weight to terms that occur often in a sentence.

*Global Term Weighting (*isf). The inverse sentence frequency (isf) measures the importance of a term within the sentence collection:

$$\mathrm{isf}_{ik} = \log\left(\frac{n}{n_k}\right). \tag{2}$$

Here n is the number of all sentences in the collection, and n_k is the number of sentences that term t_k occurs in sentence s_i. A term that occurs in every sentence of the collection gets a lower isf value. This reflects the fact that it is not as significant for the distinction between sentences as terms that occur rarely throughout the sentence collection. The isf factor has been introduced to improve the discriminating power of terms in the traditional IR.

This results in the tf-isf weighting scheme:

$$w_{ik} = \mathrm{tf}_{ik} \times \mathrm{isf}_k = \mathrm{tf}_{ik} \times \log\left(\frac{n}{n_k}\right), \tag{3}$$

where the weight w_{ik} of a term t_k in a sentence s_i is defined by the product of the local weight of term t_k in sentence s_i and the global weight of term t_k.

Similarity Calculation. A very popular similarity measure is the cosine similarity. The cosine similarity uses the weighting terms representation of the sentences. According to the VSM the sentence s_i represented as a weighting vector of the terms, $s_i = [w_{i1}, w_{i2}, \ldots, w_{im}]$, where w_{ik} is the weight of the term t_k in the sentence s_i. This measure is based on the angle α between two vectors in the VSM. The closer the vectors are to each other the more similar are the sentences. The calculation of an angle between two vectors $s_i = [w_{i1}, w_{i2}, \ldots, w_{im}]$ and $s_j = [w_{j1}, w_{j2}, \ldots, w_{jm}]$ can be derived from the Euclidean dot product:

$$\left(s_i, s_j\right) = |s_i| \cdot \left|s_j\right| \cdot \cos\alpha. \tag{4}$$

This states that the product of two vectors is given by the product of their norms (in spatial terms, the length of the vector) multiplied by the cosine of the angle α between them. Given (4) the cosine similarity is therefore

$$\mathrm{sim}\left(s_i, s_j\right) = \cos\alpha = \frac{\left(s_i, s_j\right)}{|s_i| \cdot \left|s_j\right|} = \frac{\sum_{k=1}^{m} w_{ik} w_{jk}}{\sqrt{\sum_{k=1}^{m} w_{ik}^2 \cdot \sum_{k=1}^{m} w_{jk}^2}}, \quad i, j = 1, 2, \ldots, n. \tag{5}$$

3.3. Mathematical Formulation of Problem. First, we introduce the following variables and notations:

(i)

$$y_j = \begin{cases} 1, & \text{if sentence } s_j \text{ is selected as median,} \\ 0, & \text{otherwise,} \end{cases} \tag{6}$$

(ii)

$$x_{ij} = \begin{cases} 1, & \text{if sentence } s_i \text{ is allocated to median, } s_j \\ 0, & \text{otherwise,} \end{cases} \tag{7}$$

(iii) $O = (o_1, o_2, \ldots, o_m)$ is the center of collection $D = \{s_1, s_2, \ldots, s_n\}$, which kth component o_k is defined as:

$$o_k = \left(\frac{1}{n}\right) \sum_{i=1}^{n} w_{ik}, \quad k = 1, 2, \ldots, m, \tag{8}$$

(iv) l_i is the length (in words or bytes) of sentence s_i,

(v) L is the length of summary.

We attempt to find a subset of the set $D = \{s_1, s_2, \ldots, s_n\}$ that covers the main content of the document collection while reducing the redundancy in the summary. If we let $S \subset D$ be the set of sentences constituting a summary, then the similarity between the set of sentences and the summary is going to be $\mathrm{sim}(S, D)$, which we would like to maximize. As already mentioned above, the offered model is based on the p-median problem. This approach is used to detect the topics of documents. Detection of topics helps to cover as much as possible themes from the documents. After topics detection to form a summary from each group should be selected such sentences that would be relevant to corresponding topic. For this purpose the informative sentences from each group, that is, median sentences, are selected. On the other hand, the summary length should not exceed the given limit. In our statement, it is supposed that the summary will be created by median sentences then at a choice of sentence as median, it is necessary to meet a condition that the sum of length of the selected median sentences will not exceed the given summary length. From [36] we know that the centre is the basic point carrying the main content of sentence collection. Therefore relevance of the summary will be defined as an affinity measure between them and the centre O of all set of sentences, $\mathrm{sim}(S, O)$.

Under the notations and statements above, text summarization task, we formalize as follows:
maximize

$$f(x, y) = \mathrm{sim}\left(\oplus_{j=1}^{n} s_j y_j, O\right) + \sum_{i=1}^{n} \sum_{j=1}^{n} \mathrm{sim}\left(s_i, s_j\right) x_{ij} - \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathrm{sim}\left(s_i, s_j\right) y_i y_j, \tag{9}$$

subject to

$$\sum_{j=1}^{n} x_{ij} = 1, \quad \forall i = 1, 2, \ldots, n, \tag{10}$$

$$x_{ij} \leq y_j, \quad \forall i, j = 1, 2, \ldots, n, \tag{11}$$

$$\sum_{j=1}^{n} y_j l_j \leq L, \tag{12}$$

$$y_j \in \{0, 1\}, \quad \forall j = 1, 2, \ldots, n, \tag{13}$$

$$x_{ij} \in \{0, 1\}, \quad \forall i, j = 1, 2, \ldots, n, \tag{14}$$

where $S = \oplus_{j=1}^{n} s_j y_j$ is the summary and $\oplus$ is the concatenation operation. Sentence concatenation is the operation of joining two sentences end to end.

The objective function (9) balances the relevance, content coverage, and diversity in the summary. The first term aims to evaluate the relevancy of the summary. Higher value of the term corresponds to higher relevancy of the summary. The second term aims to evaluate the content coverage of the summary. The high value of the term provides that sentences be well grouped in topics. As said above the summary should not contain multiple sentences that convey the same information. Since, in our formulation, it is supposed that the summary will be formed of medians then at choosing of sentences as a median, it is necessary to meet a condition that similarity between them was minimum. This requirement will be provided by the third term. Lower value of the term corresponds to higher diversity in the summary. Constraint (10) ensures that each sentence should be associated with one and only one median, while constraints (11) restrict sentences to be assigned to open medians. Constraint (12) implies that the length constraint of summary cannot be violated. Finally, constraints (13) and (14) refer to integrality constraints.

As seen, the ranges of the three terms in (9) are very different. For example, the range of the first term is $[0,1]$, whereas those of the second and third ones are $[0,n]$ and $[0,\ n\ (n-1)/2]$, respectively. Therefore, in (9) we normalize the second and third terms so that their ranges were also $[0,1]$. Thus, we get the following objective function:

$$\begin{aligned} f(x,y) = {} & \operatorname{sim}\left(\oplus_{j=1}^{n} s_j y_j, O\right) + \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\operatorname{sim}\left(s_i,s_j\right)x_{ij} \\ & - \frac{2}{n(n-1)}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\operatorname{sim}\left(s_i,s_j\right)y_i y_j. \end{aligned} \tag{9'}$$

Now, our objective is to find the binary assignments $\mathbf{Y} = [y_j]$ and $\mathbf{X} = [x_{ij}]$ with the high relevancy, best content coverage, and less redundancy such that the summary length is at most L.

4. Self-Adaptive Differential Evolution Algorithm

DE proposed by Storn and Price [37] is a fast and simple technique, which performs well on a wide variety of problems [38–40]. DE is a population-based stochastic search technique like genetic algorithm using the three operators: crossover, mutation, and selection. The main difference in constructing better solutions is that genetic algorithms rely on crossover while DE relies on mutation operation. This main operation is based on the differences of randomly sampled pairs of solutions in the population. The algorithm uses mutation operation as a search mechanism and selection operation to direct the search toward the prospective regions in the search space.

The basic idea which DE scheme is based on is to generate new trial vector. When mutation is implemented, several differential vectors obtained from the difference of several randomly chosen parameter vectors are added to the target vector to generate a mutant vector. Then, a trial vector is produced by crossover recombining the obtained mutant vector with the target vector. Finally, if the trial vector yields better fitness value than the target vector, replace the target vector with the trial vector [37–40].

In this section, a self-adaptive DE (SaDE) algorithm is created to solve the optimization problem (9′)–(14). The main steps of the SaDE algorithm are described below.

4.1. Initialization of the Parameter Vectors.

DE searches for a global optimum point an n-dimensional real parameter space. It begins with a randomly initiated population of NP n-dimensional real-valued parameter vectors. Each vector forms a candidate solution to the multidimensional optimization problem. We shall denote subsequent generations in DE by $t = 0,1,\ldots,t_{\max}$. Since the parameter vectors are likely to be changed over different generations, we adopt the following notation for representing the pth vector of the population at the current t generation: $X_p(t) = [x_{p,1}(t), x_{p,2}(t),\ldots,x_{p,n}(t)]$, where $x_{p,i}(t)$ is the ith component of the pth vector in the population, $i = 1,2,\ldots,n$, $p = 1,2,\ldots,\mathrm{NP}$ [38].

For each parameter of the problem, there may be a certain range within which the value of the parameter should be restricted, often because parameters are related to physical components or measures that have natural bounds. The initial population (at $t = 0$) should cover the entire search space as much as possible by uniformly randomizing the initial individuals within the search space constrained by the prescribed minimum and maximum parameter bounds $X^{\min} = [x_1^{\min},\ldots,x_n^{\min}]$ and $X^{\max} = [x_1^{\max},\ldots,x_n^{\max}]$. For example, the initial value of the ith component of the pth individual $X_p(t)$ at generation $t = 0$ is generated by [38]

$$x_{p,i}(0) = x_i^{\min} + \left(x_i^{\max} - x_i^{\min}\right)\cdot \mathrm{rand}_{p,i}, \tag{15}$$

where $\mathrm{rand}_{p,i}$ is a uniformly distributed random number lying between 0 and 1 and is instantiated independently for each component $i \in \{1,2,\ldots,n\}$ of the pth vector.

4.2. Mutation with Self-Adaptive Scaling Factor.

After initialization, DE employs the mutation operation to produce a mutant vector with respect to each individual, so-called target vector, in the current generation. For each target vector, $X_p(t)$, at generation t, its associated mutant vector $Y_p(t)$ can be generated by following strategy:

$$Y_p(t) = X_{p1}(t) + F\cdot\left(X_{p2}(t) - X_{p3}(t)\right), \tag{16}$$

where $X_{p1}(t)$, $X_{p2}(t)$, and $X_{p3}(t)$ are randomly chosen individuals from the same generation with $p \neq p1 \neq p2 \neq p3$ and the F is a scaling factor which is a positive constant. The scaling factor F is used to effect the amplification of the difference vector, $X_{p2}(t) - X_{p3}(t)$. A general setting for this factor is $F \in [0,2]$. However, Storn and Price [37] suggest $F \in [0.5,1]$ as such a setting may result in good optimization effectiveness.

The scaling factor F that is set by the user is generally a key factor affecting the DE's performance. Choosing suitable value of F is difficult for DE, which is usually problem dependent. Therefore, since introduction, a large body of research has been done to study the performance of DE and to improve its performance. In recent years, many approaches are attempted to improve the performance of DE by variable scaling factor [37–41]. The scaling factor is critical for the performance of DE, which balances global exploration and local exploitation abilities of the population. A large mutation factor facilitates exploration, but it takes the population long time to converge. Conversely, a small scaling factor makes the population fast converge, but it sometimes leads to local optimal. Hence, the adaptive and self-adaptive DE algorithms are proposed in the literature [38–41]. Furthermore, introducing the same scaling factor for all individuals, by ignoring the differences among individuals' performances, is not a precise model. In fact, during the search every individual dynamically changes its position, so every individual locates in a complex environment and faces a different situation. Therefore, every individual may have different tradeoffs between global and local search abilities.

Motivated by what is previously mentioned in this section, the scaling factor is dynamically adapted for every individual by introducing a measure called affinity index, which characterizes the nearness of personal solution of individuals to the global solution of population at the tth iteration. Based on this index, every individual could decide how to adjust the values of scaling factor. For this purpose, the mutation strategy is given by

$$Y_p(t) = X_p(t) + F_p(t) \cdot \left(X_{p1}(t) - X_{p2}(t)\right). \tag{17}$$

To calculate the scaling factor for pth target vector in tth iteration, denoted by $F_p(t)$ in (17), first the affinity index (AI) is defined as follows:

$$\mathrm{AI}_p(t) = \frac{\mathrm{fit}\left(X_p(t)\right) - \mathrm{fit}(X_{\mathrm{worst}}(t))}{\mathrm{fit}(X_{\mathrm{best}}(t)) - \mathrm{fit}(X_{\mathrm{worst}}(t))}, \tag{18}$$

where $X_{\mathrm{best}}(t)$ and $X_{\mathrm{worst}}(t)$ represent the best and worst solution of the population at the iteration t, $\mathrm{fit}(X_{\mathrm{best}}(t)) = \max_{p\in\{1,\ldots,\mathrm{NP}\}}\{\mathrm{fit}(X_p(t))\}$ and $\mathrm{fit}(X_{\mathrm{worst}}(t)) = \min_{p\in\{1,\ldots,\mathrm{NP}\}}\{\mathrm{fit}(X_p(t))\}$, and $\mathrm{fit}(\cdot)$ is the fitness function.

It can be concluded that a small $\mathrm{AI}_p(t)$ means that the pth individual is far away from the global best solution and it needs a strong global exploration, therefore a large perturbation. On the other hand, a big $\mathrm{AI}_p(t)$ means that pth individual has a high nearness to the global solution, and so it needs a strong local exploitation, therefore a small perturbation [38, 40]. Hence, the value of scaling factor for every target individual in tth iteration is dynamically adapted with the following formula:

$$F_p(t) = \frac{1}{1 + \tanh\left(2 \cdot \mathrm{AI}_p(t)\right)}, \tag{19}$$

where $\tanh(z)$ is the hyperbolic tangent function:

$$\tanh(z) = \frac{\exp(z) - \exp(-z)}{\exp(z) + \exp(-z)}. \tag{20}$$

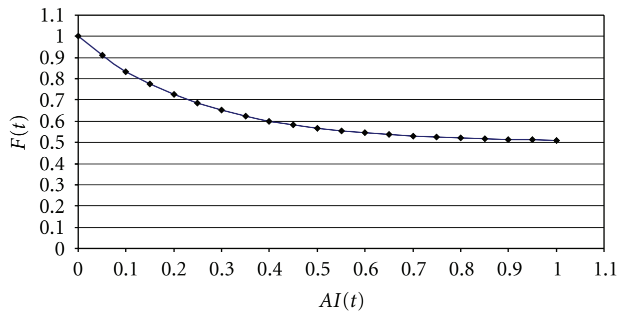

Figure 1: Scaling factor (F) versus affinity index ($\mathrm{AI}(t)$).

Under the assumption and definitions above, it can be concluded that $0.5 < F_p(t) \le 1$. Figure 1 depicts the variation of scaling factor (F) versus affinity index ($\mathrm{AI}(t)$).

According to (18) and (19), during the search, the individuals get different values of $\mathrm{AI}_p(t)$ and then scaling factor depending on their fitness. While the fitness of an individual is far away from the fitness of the global best, $\mathrm{AI}_p(t)$ for this individual has a small value and the value of scaling factor will be large resulting in strong global search abilities and locate the promising search areas. Meanwhile, when the fitness of an individual achieves near the global best, $\mathrm{AI}_p(t)$ for this individual has a big value and scaling factor will be set small, depending on the neighbor of its best fitness to the global best value, to facilitate a finer local explorations and so accelerate convergence.

4.3. Crossover. To enhance the potential diversity of the population, a crossover operation comes into play after generating the mutant vector through mutation. The mutant vector mixes its components with the target vector $X_p(t)$ under this operation to form the trial vector $Z_p(t)$. The DE family of algorithms can use two kinds of crossover methods—exponential (or two-point modulo) and binomial (or uniform) [37, 38]. In this paper we focus on the widely used binomial crossover that is performed on each of the n variables whenever a randomly generated number between 0 and 1 is less than or equal to the CR value. In this case, the number of parameters inherited from the mutant vector has a (nearly) binomial distribution. The scheme may be outlined as

$$z_{p,i}(t) = \begin{cases} y_{p,i}(t), & \text{if } \mathrm{rand}_{p,i} \le \mathrm{CR} \text{ or } i = i_{\mathrm{rand}}, \\ x_{p,i}(t), & \text{otherwise}, \end{cases} \tag{21}$$

where $y_{p,i}(t)$ and $z_{p,i}(t)$ are the ith-dimensional components of the vectors $Y_p(t)$ and $Z_p(t)$, respectively; CR is the predefined crossover probability which is usually set to a fixed value in $(0, 1)$ or changes dynamically within $(0, 1)$; i_{rand} is a number randomly selected from the index set $\{1, 2, \ldots, n\}$ and used to ensure that the trial vector $Z_p(t)$ is different from the original solution $X_p(t)$. The crossover rate CR controls the recombination of target vector and mutant vector to generate trial vector.

If the values of some parameters of a newly generated trial vector exceed the corresponding upper and lower bounds, then all the components of the trial vector are checked whether they violate the boundary constraints. If the ith component $z_{p,i}(t)$ of the mutant vector $Z_p(t)$ violates the boundary constraint (15), $z_{p,i}(t)$ is reflected back from the violated boundary constraint as follows:

$$z_{p,i}(t) = \begin{cases} 2x_i^{\min} - z_{p,i}(t), & \text{if } z_{p,i}(t) < x_i^{\min}, \\ 2x_i^{\max} - z_{p,i}(t), & \text{if } z_{p,i}(t) > x_i^{\max}, \\ z_{p,i}(t), & \text{otherwise.} \end{cases} \quad (22)$$

4.4. Selection. The next step of the algorithm calls for selection to determine whether the target or the trial vector survives to the next generation, that is, at $t = t + 1$. Then the objective function values of all trial vectors are evaluated. After that, a selection operation is performed. The DE algorithm uses a greedy selection. The selection operator chooses between the target and corresponding trial vectors. A member of the next generation becomes the fittest vector, that is, vector with the better fitness value. For example, if we have a maximization problem, the selection operation can be expressed as follows:

$$X_p(t+1) = \begin{cases} Z_p(t), & \text{if } f\left(Z_p(t)\right) \geq f\left(X_p(t)\right), \\ X_p(t), & \text{otherwise.} \end{cases} \quad (23)$$

Therefore, if the trial vector yields an equal or better value of the objective function, it replaces the corresponding target vector in the next generation; otherwise the target is retained in the population. Hence, the population either gets better (with respect to the maximization of the objective function) or remains the same in fitness status, but never deteriorates.

4.5. Binarization. Binary DE is the modified version of DE, which operates in binary search spaces. In the binary DE, the real value of genes is converted to the binary space by the rule [42]

$$x_{p,i}(t+1) = \begin{cases} 1, & \text{if } \text{rand}_{p,i} < \text{sigm}\left(x_{p,i}(t+1)\right), \\ 0, & \text{otherwise,} \end{cases} \quad (24)$$

where, as before, $\text{rand}_{p,i}$ is a uniformly distributed random number lying between 0 and 1, which is instantiated independently for each ith component of the pth parameter vector. $\text{sigm}(z)$ is the sigmoid function:

$$\text{sigm}(z) = \frac{1}{1+\exp(-z)}. \quad (25)$$

The motivation to use the sigmoid function (25) is to map interval $[x_i^{\min}, x_i^{\max}]$ for each $i \in \{1, 2, \ldots, n\}$ into the interval $(0, 1)$, which is equivalent to the interval of a probability function. After such transformation from the real-coded representation (15) we obtain the binary-coded representation, $x_{p,i}(t) \in \{0, 1\}$, where the $x_{p,i}(t) = 1$ indicates that the ith sentence is selected to be included to the summary, otherwise, the ith sentence will not be selected. For example, the individual $X_p(t) = [1, 0, 0, 1, 1]$ represents a candidate solution that first, fourth, and fifth sentences are selected to be included to the summary.

4.6. Constraint Handling. When population initialization, mutation, crossover, and binarization have been implemented, the new generated solution may not satisfy the constraint (12). The most popular constraint handling strategy at present is penalty method, which often uses function to convert a constrained problem into an unconstraint one [43]. Therefore, this strategy is very convenient to handle the constraints for evolutionary algorithm by punishing the infeasible solution during the selection procedure to ensure the feasible ones are favored. However, this strategy has some drawbacks and the main one is the requirement of multiple runs for the fine-tuning of penalty factors, which would increase the computational time and degrade the efficiency of the algorithm. In order to overcome the drawbacks of penalty method and handle the constraints of problem effectively, the following heuristic procedure is produced for all NP solutions in the population to resolve the constraint (12).

Constraint is handled by using a suitable fitness function, which depends on the current population. Solutions in a population are assigned fitness so that feasible solutions are emphasized more than infeasible solutions. The following three criteria are satisfied during the handling process.

(i) Any feasible solution wins over any infeasible solution.

(ii) Two feasible solutions are compared only based on their objective function values.

(iii) Two infeasible solutions are compared based on the amount of constraint violation.

4.7. Stopping Criterion. Mutation, crossover and selection continue until some stopping criterion is satisfied. If the predefined maximum iteration number is reached, then the DE algorithm is terminated and outputs the best solution obtained by DE as the result. Otherwise, it is continued to carry out individual's position updates process (mutation, crossover, and selection process).

4.8. Parameter Settings of SaDE Algorithm. The crucial parameters that affect the performance of DE are the population size (NP), crossover rate (CR), and the scaling parameter (F). In the proposed SaDE algorithm the population size, NP $= 50$, is maintained constant throughout the evolution process. This is a heuristic choice. For example, the value of NP can be increased for obtaining the global solution with higher probability. However, the higher the value of NP is, the higher the number of fitness evaluation is. The crossover rate controls which and how many components are mutated in each element of the current population. The crossover rate CR is a probability of mixing between trial and target vectors. A large CR often speeds up convergence. However, from a certain value upwards, the convergence speed may decrease or the population may converge prematurely. In [40], a good choice for CR is said to be between 0.3 and 0.9. Therefore we ran self-adaptive DE for CR $= 0.9, 0.8, 0.7, 0.6, 0.5, 0.4$, and 0.3. The best results presented for DE are obtained for CR $=$ 0.9. Other parameters are set as $t_{\max} = 1000$, $x_i^{\min} = -5$, and $x_i^{\max} = 5$ for all $i \in \{1, 2, \ldots, n\}$.

4.9. Framework of the SaDE Algorithm. The pseudocode of the proposed SaDE algorithm can be summarized as follows.

Step 1 (initial population). Using the rule (15) create an initial population.

Step 2 (binarization). Transform real-coded individuals to binary-coded individuals using (24).

Step 3 (select best and worst individuals). Select the individuals with current best and worst solution.

Step 4 (generate trial vector). Generate trial vector of the target vector using the mutation (17)–(20) and crossover (21) operators.

Step 5 (check the boundary constraints). Check the boundary constraints (15) for the components of the trial vector using (22).

Step 6 (binarization). Transform real-coded trial vector to binary-coded trial vector using (24).

Step 7 (constraint handling). Handle the constraint (12) using the strategy described in Section 4.6.

Step 8 (selection). If a trial vector is better than its target vector is, then replace the target vector by trial vector in the next generation.

Step 9 (stopping criterion). Repeat Steps 2–8 until a user-specified maximum number $t_{\max}$ of fitness calculation is reached.

Step 10 (output). Report the summary obtained by the best individual as the final solution at maximum number of fitness calculation.

5. Experimental Results

5.1. Dataset. We take the DUC 2006 data set as the evaluation corpora [44]. DUC 2006 provides 50 document sets for evaluation. Each document set includes a fixed number—25 documents. For each topic, four human summarizers are asked to provide a 250-word summary of the topic from the 25 related documents for automatic evaluation. All documents are preprocessed by removing stop words [45] and conducting stemming [46].

5.2. Evaluation Metrics. A well-recognized automatic evaluation toolkit ROUGE [47] is used in evaluation. It includes five measures, which automatically determine the quality of a machine-generated summary by comparing it to ideal summaries created by humans: ROUGE-N, ROUGE-L, ROUGE-W, ROUGE-S, and ROUGE-SU. These measures evaluate the quality of the summarization by counting the number of overlapping units, such as N-grams, between the generated summary by a method and a set of reference summaries.

The ROUGE-N measure compares N-grams of two summaries and counts the number of matches:

$$\text{ROUGE-}N = \frac{\sum_{S\in \text{Summ}_{\text{ref}}}\sum_{N\text{-gram}\in S}\text{Count}_{\text{match}}(N\text{-gram})}{\sum_{S\in \text{Summ}_{\text{ref}}}\sum_{N\text{-gram}\in S}\text{Count}(N\text{-gram})}, \tag{26}$$

where N stands for the length of the N-gram, $\text{Count}_{\text{match}}(N\text{-gram})$ is the maximum number of N-grams co-occurring in candidate summary and the set of reference-summaries. Count(N-gram) is the number of N-grams in the reference summaries.

ROUGE-L computes the ratio between the length of the summaries' longest common subsequence (LCS) and the length of the reference summary:

$$\begin{aligned} P_{\text{LCS}}(R,S) &= \frac{\text{LCS}(R,S)}{|S|},\\ R_{\text{LCS}}(R,S) &= \frac{\text{LCS}(R,S)}{|R|},\\ F_{\text{LCS}}(R,S) &= \frac{(1+\beta^2)P_{\text{LCS}}(R,S)R_{\text{LCS}}(R,S)}{\beta^2 P_{\text{LCS}}(R,S)+R_{\text{LCS}}(R,S)}, \end{aligned} \tag{27}$$

where $|R|$ and $|S|$ are the length of the reference R and candidate S sentence summaries, respectively. LCS(R,S) is the length of an LCS of R and S. $P_{\text{LCS}}(R,S)$ is the precision of LCS(R,S), $R_{\text{LCS}}(R,S)$ is the recall of LCS(R,S), and $\beta = P_{\text{LCS}}(R,S)/R_{\text{LCS}}(R,S)$.

Lin [47] implemented two extensions to ROUGE-N: skip-bigram cooccurrence (ROUGE-S) and skip-bigram co-occurrence averaged with unigram cooccurrence (ROUGE-SU). The way ROUGE-S is calculated is identical to that of ROUGE-2, except that skip bigrams are defined as subsequences rather than the regular definition of bigrams as substrings. Skip-bigram (skip bigram is any pair of words in their sentence order, allowing for arbitrary gaps) cooccurrence statistics, ROUGE-S, measure the similarity of a pair of summaries based on how many skip bigrams they have in common:

$$\begin{aligned} P_{\text{SKIP2}}(R,S) &= \frac{\text{SKIP2}(R,S)}{C(|S|,2)},\\ R_{\text{SKIP2}}(R,S) &= \frac{\text{SKIP2}(R,S)}{C(|R|,2)},\\ F_{\text{SKIP2}}(R,S) &= \frac{(1+\beta^2)P_{\text{SKIP2}}(R,S)R_{\text{SKIP2}}(R,S)}{\beta^2 P_{\text{SKIP2}}(R,S)+R_{\text{SKIP2}}(R,S)}, \end{aligned} \tag{28}$$

where SKIP2(R,S) is the number of skip-bigram matches between R and S, β is the relative importance of $P_{\text{SKIP2}}(R,S)$ and $R_{\text{SKIP2}}(R,S)$, $P_{\text{SKIP2}}(R,S)$ being the precision of SKIP2(R,S) and $R_{\text{SKIP2}}(R,S)$ the recall of SKIP2(R,S). $C(\cdot,\cdot)$ is the combination function.

One potential problem for ROUGE-S is that it does not give any credit to a candidate sentence if the sentence does not have any word pair co-occurring with its references. To accommodate this, ROUGE-S is extended with the addition of unigram as counting unit. The extended version is called ROUGE-SU that is a weighted average between ROUGE-S and ROUGE-1.

Table 1: ROUGE scores of the methods.

Methods	ROUGE-1	ROUGE-2	ROUGE-SU4
pSum-SaDE	**0.4416**	**0.0994**	**0.1592**
LEX	0.4030	0.0913	0.1449
TMR + TF	0.4063	0.0913	0.1504
HybHSum	0.4300	0.0910	0.1510
PLSA-JS	0.4328	0.0970	0.1557
iRANK	0.4032	0.0912	0.1450
HierSum	0.4010	0.0860	0.1430
SVR	0.4018	0.0926	0.1485

Table 2: Comparison pSum-SaDE with other methods.

Methods	Improvement of the method pSum-SaDE, %		
	ROUGE-1	ROUGE-2	ROUGE-SU4
LEX	9.58	8.87	9.87
TMR + TF	8.69	8.87	5.85
HybHSum	2.70	9.23	5.43
PLSA-JS	2.03	2.47	2.25
iRANK	9.52	8.99	9.79
HierSum	10.12	15.58	11.33
SVR	9.91	7.34	7.21
Average	*7.51*	*8.77*	*7.39*

5.3. Performance Evaluation. In this section, we empirically compare the systems using ROUGE-1, ROUGE-2, and ROUGE-SU4 metrics. ROUGE-SU4 is an extended version of ROUGE-2 that allows word-level gaps of maximum length 4 between the bigram tokens. These metrics were computed by comparing automatically generated summaries against the model summaries. We implement the following most widely used document summarization methods as the baseline systems to compare with our method: LEX [33], TMR + TF [22], HybHSum [26], PLSA-JS [25], iRANK [24], HierSum [23], and SVR [21]. Table 1 provides the ROUGE scores of the methods. The results reported here are averaged over 20 runs. In Table 1 through pSum-SaDE our method is denoted.

From the results reported in Table 1, we have the following observations: (1) the method pSum-SaDE outperforms all other methods and its results are close to results of the method PLSA-JS; (2) HierSum has the worst performance; (3) the results of LEX and iRANK are similar.

Table 2 demonstrates the improvements of the method pSum-SaDE in all ROUGE scores. We clearly observe that our method achieves the highest ROUGE scores and outperforms all the other systems. For comparison we have used the relative improvement ((our method − other methods)/other methods) × 100. We also observe that among other methods the PLSA-JS shows the best results compared to other methods. Compared with the method PLSA-JS the method pSum-SaDE improves the performance by 2.03%, 2.47%, and 2.25% in terms ROUGE-1, ROUGE-2, and ROUGE-SU4 metrics, respectively.

Table 3: Comparison with the three best systems on DUC 2006.

Method	ROUGE-1	ROUGE-2	ROUGE-SU4
pSum-SaDE	0.4416	0.0994	0.1592
System24	0.4106	0.0951	0.1549
System15	0.4020	0.0903	0.1468
System12	0.4040	0.0890	0.1469

We further compared our results with the three best participant systems [22] in DUC 2006. Table 3 shows the comparison results. We see that our proposed method outperforms the three systems. We need note that our method does not make use of any external information, while the systems usually (heavily) depend on some external knowledge, for example, System 15 employs WordNet for discovering semantic similarity between words and System 24 employs linguistic features such as named entity, linguistic patterns, and semantic similarity between words.

5.4. Comparison with Canonical DE Algorithm. This section demonstrates the feasibility of the SaDE-based document summarization. The results are compared to the results obtained from canonical DE. In SaDE and canonical DE the parameter CR does not change during search process which is equal to $CR = 0.6$. In canonical DE algorithm, the mutation strategy is defined by (16), whose parameter F does not change during search process. In our experiment, the scaling factor F is set to 0.5. To perform a fair comparison, the same computational effort is used in both of canonical DE and SaDE. That is, the maximum generation, population size, and searching range of the parameters in DE are the same as those in adaptive DE. In addition, notice that random number generator is initialized with the same seed values. The maximum generation, the population size, and the maximum number of iterations are set to 20, 50, and 1000, respectively.

Table 4 shows the worst, mean, best, and standard deviation of ROUGE results during 20 runs for each algorithm DE and SaDE. From Table 4, it is obvious that the worst results obtained by adaptive DE are even better than the best results obtained by DE.

5.5. Comparison of Runtime of the Methods. In this section, CPU runtimes of the seven tested methods are compared. All the methods were implemented in the Delphi 7 language. The algorithms were run on a Server running Windows Vista with two dual-core Intel Xeon CPU (4 GHz) processors and 4 GB memory. Table 5 shows the comparison in terms of time spent by each method. Last column shows the ranks of the method on their time spent. From the experimental results, we clearly observe that (1) LEX method performs slowly; (2) the methods iRANK, HierSum, and HybHSum, and the methods PLSA-JS and SVR spend almost equal time; (3) our methods pSum-DE and pSum-SaDE take the first and second places, respectively.

5.6. Statistical Significance Test. To judge the statistical significance of the summarization results, a nonparametric

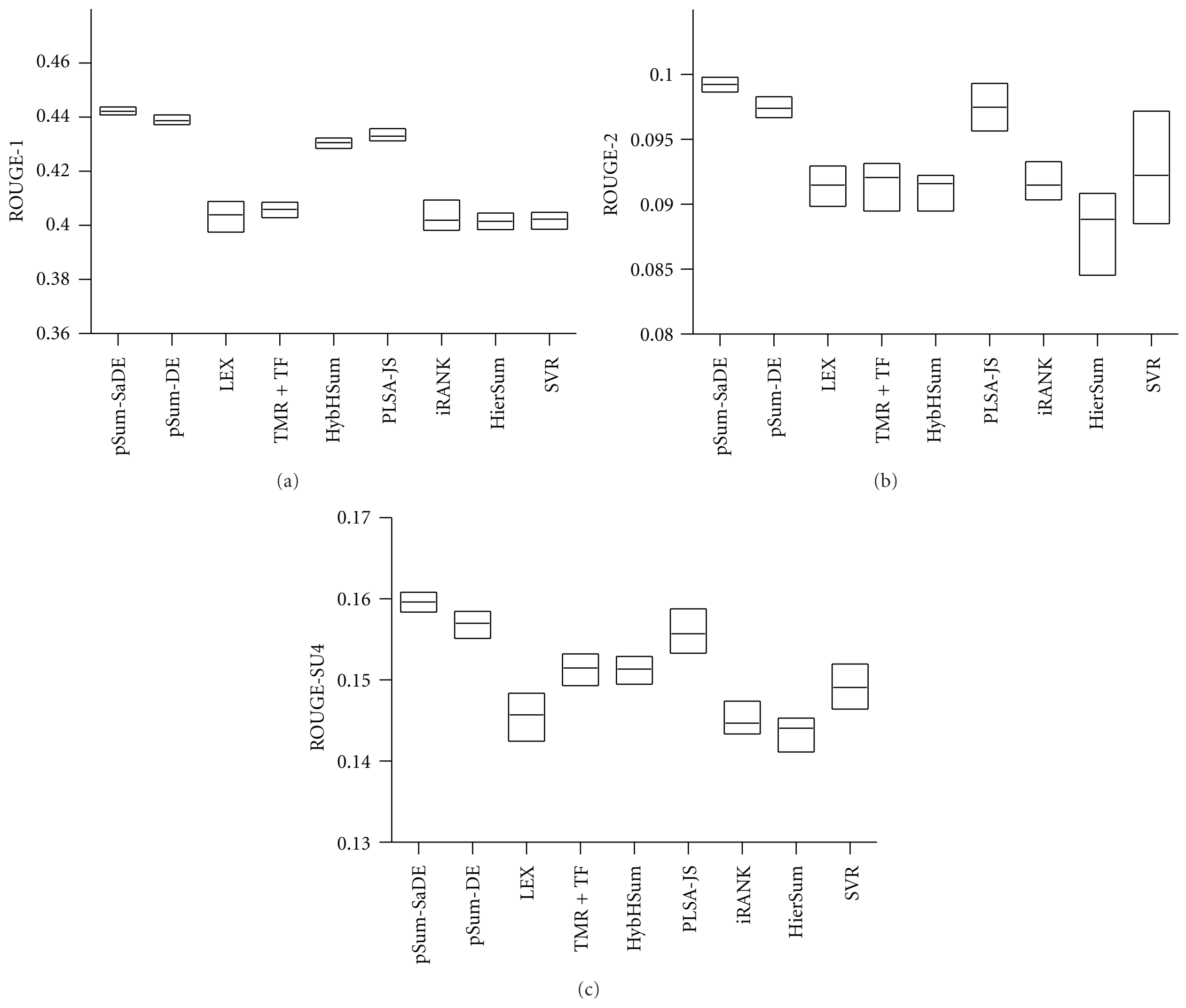

Figure 2: Change of ROUGE-1, ROUGE-2, and ROUGE-SU4 for different summarization methods.

Table 4: Comparison pSum-SaDE with pSum-DE.

	Algorithm	Worst	Mean	Best	Stdv
ROUGE-1	SaDE	0.4399	**0.4416**	0.4428	**1.13*e* − 04**
	DE	0.4366	0.4378	0.4388	1.45*e*− 04
ROUGE-2	SaDE	0.0987	**0.0994**	0.1011	**1.11*e* − 04**
	DE	0.0955	0.0967	0.0975	1.47*e*− 04
ROUGE-SU4	SaDE	0.1584	**0.1592**	0.1605	**1.01*e* − 04**
	DE	0.1537	0.1564	0.1575	1.36*e*− 04

Table 5: Comparison of the methods on time spent.

Methods	Time (min)	Rank
pSum-DE	42.7	1
pSum-SaDE	44.2	2
TMR + TF	51.3	3
PLSA-JS	54.4	4
SVR	56.6	5
iRANK	62.7	6
HierSum	63.1	7
HybHSum	64.9	8
LEX	67.6	9

statistical significance test called Wilcoxon's rank sum test for independent samples [48] has been conducted at the 5% significance level. Nine groups, corresponding to the nine methods ((1) pSum-SaDE, (2) pSum-DE, (3) LEX, (4) TMR + TF, (5) HybHSum, (6) PLSA-JS, (7) iRANK, (8) HierSum, (9) SVR), have been created for each data set. Two groups are compared at a time one corresponding to pSum-SaDE method and the other corresponding to some other method considered in this paper. Each group consists of the ROUGE scores for the data sets produced by 20 consecutive runs of the corresponding method. The median values, 95% CI, and standard error (stdr.) of ROUGE scores of each group for all the data sets are shown in Table 6.

Table 6: Median values, 95% CI, and standard error of ROUGE scores over 20 consecutive runs of methods.

Methods	ROUGE-1			ROUGE-2			ROUGE-SU4		
	Median	95% CI	Stdr.	Median	95% CI	Stdr.	Median	95% CI	Stdr.
pSum-SaDE	0.4422	[0.4417, 0.4426]	2.2$e-4$	0.0992	[0.0990, 0.0994]	7.8$e-5$	0.1596	[0.1591, 0.1598]	1.7$e-4$
pSum-DE	0.4387	[0.4383, 0.4394]	2.5$e-4$	0.0974	[0.0971, 0.0976]	1.1$e-4$	0.1570	[0.1564, 0.1575]	2.6$e-4$
LEX	0.4040	[0.4022, 0.4057]	8.3$e-4$	0.0915	[0.0910, 0.0919]	2.3$e-4$	0.1457	[0.1448, 0.1465]	4.1$e-4$
TMR + TF	0.4059	[0.4048, 0.4067]	4.4$e-4$	0.0921	[0.0911, 0.0922]	2.7$e-4$	0.1515	[0.1508, 0.1520]	3.0$e-4$
HybHSum	0.4306	[0.4299, 0.4310]	2.6$e-4$	0.0916	[0.0908, 0.0916]	2.1$e-4$	0.1514	[0.1508, 0.1518]	2.6$e-4$
PLSA-JS	0.4330	[0.4327, 0.4341]	3.2$e-4$	0.0975	[0.0970, 0.0981]	2.6$e-4$	0.1557	[0.1549, 0.1565]	4.0$e-4$
iRANK	0.4019	[0.4014, 0.4046]	7.8$e-4$	0.0915	[0.0912, 0.0920]	2.0$e-4$	0.1447	[0.1446, 0.1458]	2.9$e-4$
HierSum	0.4016	[0.4006, 0.4023]	4.2$e-4$	0.0889	[0.0877, 0.0894]	4.1$e-4$	0.1441	[0.1427, 0.1442]	3.5$e-4$
SVR	0.4024	[0.4008, 0.4030]	5.1$e-4$	0.0922	[0.0915, 0.0941]	6.1$e-4$	0.1491	[0.1483, 0.1497]	3.4$e-4$

Table 7: P values produced by Wilcoxon's rank sum test by comparing pSum-SaDE with other methods.

Metric	P values (comparing medians of pSum-SaDE with other methods)							
	pSum-DE	LEX	TMR + TF	HybHSum	PLSA-JS	iRANK	HierSum	SVR
ROUGE-1	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001
ROUGE-2	<0.0001	<0.0001	<0.0001	<0.0001	0.0002	<0.0001	<0.0001	<0.0001
ROUGE-SU4	<0.0001	<0.0001	<0.0001	<0.0001	0.0001	<0.0001	<0.0001	<0.0001

As is evident from Table 6, the median values of ROUGE-1, ROUGE-2, and ROUGE-SU4 scores for pSum-SaDE are better than those for the other methods. To establish that this goodness is statistically significant, Table 7 reports the P values produced by Wilcoxon's rank sum test for comparison of two groups (one group corresponding to pSum-SaDE and another group corresponding to some other algorithm) at a time [49]. As a null hypothesis, it is assumed that there are no significant differences between the median values of two groups, whereas the alternative hypothesis is that there is significant difference in the median values of the two groups. It is clear from the table that P values are much less than 0.05 (5% significance level). This is strong evidence against the null hypothesis, indicating that the better median values of the performance metrics produced by pSum-SaDE are statistically significant and have not occurred by chance. Similar results are obtained for all other data sets and for all other methods compared to pSum-SaDE method, establishing the significant superiority of the proposed technique.

The superiority of pSum-SaDE method is also evident from Figure 2 that provides the range of solutions obtained by the different methods [49].

6. Conclusion and Future Work

The main contributions of the paper are the following.

(i) The paper presents a document summarization model which extracts salient sentences from given documents while reducing redundant information in the summaries with the coverage of latent topics of document collection.

(ii) Document summarization is formalized as a modified p-median problem that takes into account four basic requirements, namely, relevance, information coverage, diversity, and length limit that should satisfy summaries.

(iii) To solve the modified p-median problem a self-adaptive differential evolution algorithm is created. In particular, in the paper is proposed a self-adaptive scaling factor in original DE to increase the exploration and exploitation ability.

(iv) This paper has found that self-adaptive differential evolution can efficiently find the best solution in comparison with the canonical differential evolution. Experimental results on DUC 2006 dataset have shown that our optimization approach compares well to several summarization methods.

There are several possible directions of future research. One of them involves replacement of the SaDE algorithm in pSum-SaDE with a better global search method, such as particle swarm optimization. Another direction of future research is related to the different combination of the three terms, namely, relevancy, coverage, and redundancy terms in objective function (9′).

Acknowledgments

The authors thank the Professor Chuan-Kang Ting (the editor), and the reviewers providing very helpful comments and suggestions. Their insight and comments led to a better presentation of the ideas expressed in this paper.

References

[1] C. C. Yang and F. L. Wang, "Hierarchical summarization of large documents," *Journal of the American Society for Information Science and Technology*, vol. 59, no. 6, pp. 887–902, 2008.

[2] H. Dong, S. Yu, and Y. Jiang, "Text mining on semi-structured e-government digital archives of China," in *Proceedings of the 2nd Pacific-Asia Conference on Web Mining and Web-Based Application (WMWA '09)*, pp. 11–14, Wuhan, China, June 2009.

[3] M. A. Fattah and F. Ren, "GA, MR, FFNN, PNN and GMM based models for automatic text summarization," *Computer Speech and Language*, vol. 23, no. 1, pp. 126–144, 2009.

[4] I. Mani and M. T. Maybury, *Advances in Automatic Text Summarization*, MIT Press, Cambridge, UK, 1999.

[5] J. Otterbacher, G. Erkan, and D. R. Radev, "Biased LexRank: passage retrieval using random walks with question-based priors," *Information Processing and Management*, vol. 45, no. 1, pp. 42–54, 2009.

[6] D. Shen, J.-T. Sun, H. Li, Q. Yang, and Z. Chen, "Document summarization using conditional random fields," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pp. 2862–2867, Hyderabad, India, 2007.

[7] X. Wan, "Using only cross-document relationships for both generic and topic-focused multi-document summarizations," *Information Retrieval*, vol. 11, no. 1, pp. 25–49, 2008.

[8] Y. Ouyang, W. Li, S. Li, and Q. Lu, "Applying regression models to query-focused multi-document summarization," *Information Processing and Management*, vol. 47, no. 2, pp. 227–237, 2011.

[9] X. Wan, "An exploration of document impact on graph-based multi-document summarization," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 755–762, Honolulu, Hawaii, USA, 2008.

[10] M. Kutlu, C. Ciğir, and I. Cicekli, "Generic text summarization for Turkish," *Computer Journal*, vol. 53, no. 8, pp. 1315–1323, 2010.

[11] J. Carbonell and J. Goldstein, "The use of MMR, diversity-based reranking for reordering documents and producing summaries," in *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 335–336, Melbourne, Australia, 1998.

[12] M. S. Binwahlan, N. Salim, and L. Suanmali, "MMI diversity based text summarization," *International Journal of Computer Science and Security*, vol. 3, no. 1, pp. 23–33, 2009.

[13] X. Wan, J. Yang, and J. Xiao, "CollabSum: exploiting multiple document clustering for collaborative single document summarizations," in *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '07)*, pp. 143–150, Amsterdam, The Netherlands, July 2007.

[14] X. Cai, W. Li, Y. Ouyang, and H. Yan, "Simultaneous ranking and clustering of sentences: an reinforcement approach to multi-document summarization," in *Proceedings of the 23rd International Conference on Computational Linguistics*, pp. 134–142, Beijing, China, 2010.

[15] T. Nomoto and Y. Matsumoto, "The diversity-based approach to open-domain text summarization," *Information Processing and Management*, vol. 39, no. 3, pp. 363–389, 2003.

[16] S. Park, B. Cha, and D. An, "Automatic multi-document summarization based on clustering and nonnegative matrix factorizationfs," *IETE Technical Review*, vol. 27, no. 2, pp. 167–178, 2010.

[17] R. M. Aliguliyev, "Clustering techniques and discrete particle swarm optimization algorithm for multi-document summarization," *Computational Intelligence*, vol. 26, no. 4, pp. 420–448, 2010.

[18] R. M. Aliguliyev, "A new sentence similarity measure and sentence based extractive technique for automatic text summarization," *Expert Systems with Applications*, vol. 36, no. 4, pp. 7764–7772, 2009.

[19] R. M. Alguliev and R. M. Aliguliyev, "Evolutionary algorithm for extractive text summarization," *Intelligent Information Management*, vol. 1, no. 2, pp. 128–138, 2009.

[20] D. Wang, S. Zhu, T. Li, Y. Chi, and Y. Gong, "Integrating clustering and multi-document summarization to improve document understanding," in *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM '08)*, pp. 1435–1436, Napa Valley, Calif, USA, October 2008.

[21] Y. Ouyang, S. Li, and W. Li, "Developing learning strategies for topic-based summarization," in *Proceedings of the 16th ACM Conference on Information and Knowledge Management (CIKM '07)*, pp. 79–86, Lisbon, Portugal, November 2007.

[22] J. Tang, L. Yao, and D. Chen, "Multi-topic based query-oriented summarization," in *Proceedings of the 9th SIAM International Conference on Data Mining (SDM '09)*, pp. 1141–1152, Sparks, Nev, USA, May 2009.

[23] A. Haghighi and L. Vanderwende, "Exploring content models for multi-document summarization," in *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 362–370, Boulder, Colo, USA, 2009.

[24] F. Wei, W. Li, and S. Liu, "iRANK: a rank-learn-combine framework for unsupervised ensemble ranking," *Journal of the American Society for Information Science and Technology*, vol. 61, no. 6, pp. 1232–1243, 2010.

[25] L. Hennig, "Topic-based multi-document summarization with probabilistic latent semantic analysis," in *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, pp. 144–149, Borovets, Bulgaria, 2009.

[26] A. Celikyilmaz and D. Hakkani-Tur, "A hybrid hierarchical model for multi-document summarization," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 815–824, Uppsala, Sweden, 2010.

[27] E. Filatova and V. Hatzivassiloglou, "A formal model for information selection in multi-sentence text extraction," in *Proceedings of the 20th International Conference on Computational Linguistics*, pp. 397–403, Geneva, Switzerland, 2004.

[28] H. Takamura and M. Okumura, "Text summarization model based on maximum coverage problem and its variant," in *Proceedings of the 12th Conference of the European Chapter of the ACL*, pp. 781–789, Athens, Greece, 2009.

[29] R. McDonald, "A study of global inference algorithms in multi-document summarization," in *Proceedings of the 29th European Conference on IR Research*, LNCS, no. 4425, pp. 557–564, Springer, Rome, Italy, 2007.

[30] D. Wang, T. Li, S. Zhu, and C. Ding, "Multi-document summarization using sentence-based topic models," in *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP '09)*, pp. 297–300, Singapore, August 2009.

[31] Y. Tao, S. Zhou, W. Lam, and J. Guan, "Towards more effective text summarization based on textual association networks," in *Proceedings of the 4th International Conference on Semantics, Knowledge, and Grid (SKG '08)*, pp. 235–240, Beijing, China, December 2008.

[32] H. Takamura and M. Okumura, "Text summarization model based on the budgeted median problem," in *Proceedings of the 18th ACM International Conference on Information and Knowledge Management (CIKM '09)*, pp. 1589–1592, Hong Kong, November 2009.

[33] L. Huang, Y. He, F. Wei, and W. Li, “Modeling document summarization as multi-objective optimization,” in *Proceedings of the 3rd International Symposium on Intelligent Information Technology and Security Informatics*, pp. 382–386, Jinggangshan, China, 2010.

[34] G. Salton, “Mathematics and information retrieval,” *Journal of Documentation*, vol. 35, no. 1, pp. 1–29, 1979.

[35] J. Geiß, “Latent semantic sentence clustering for multi-document summarization,” Tech. Rep. UCAM-CL-TR-802, University of Cambridge, Computer Laboratory, Faculty of Computer Science and Technology, Cambridge, UK, 2011, http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-802.pdf.

[36] D. R. Radev, H. Jing, M. Styś, and D. Tam, “Centroid-based summarization of multiple documents,” *Information Processing and Management*, vol. 40, no. 6, pp. 919–938, 2004.

[37] R. Storn and K. Price, “Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces,” *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[38] S. Das and P. N. Suganthan, “Differential evolution: a survey of the atate-of-the-art,” *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2011.

[39] Y. Lu, J. Zhou, H. Qin, Y. Li, and Y. Zhang, “An adaptive hybrid differential evolution algorithm for dynamic economic dispatch with valve-point effects,” *Expert Systems with Applications*, vol. 37, no. 7, pp. 4842–4849, 2010.

[40] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, “Differential evolution algorithm with ensemble of parameters and mutation strategies,” *Applied Soft Computing Journal*, vol. 11, no. 2, pp. 1679–1696, 2011.

[41] L. Jia, W. Gong, and H. Wu, “An improved self-adaptive control parameter of differential evolution for global optimization,” *Communications in Computer and Information Science*, vol. 51, part 5, pp. 215–224, 2009.

[42] G. Pampara, A. P. Engelbrecht, and N. Franken, “Binary differential evolution,” in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1873–1879, Vancouver, Canada, 2006.

[43] K. Deb, “An efficient constraint handling method for genetic algorithms,” *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2–4, pp. 311–338, 2000.

[44] *Document Understanding Conference*, http://duc.nist.gov.

[45] *English Stoplist*, ftp://ftp.cs.cornell.edu/pub/smart/english.stop.

[46] *Porter Stemming Algorithm*, http://www.tartarus.org/martin/PorterStemmer/.

[47] C.-Y. Lin, “ROUGE: a package for automatic evaluation summaries,” in *Proceedings of the Workshop on Text Summarization Branches out*, pp. 74–81, Barcelona, Spain, 2004.

[48] M. Hollander and D. A. Wolfe, *Nonparametric Statistical Methods*, Wiley-Interscience, 2nd edition, 1999.

[49] *GraphPad Software*, http://www.graphpad.com/welcome.htm.