Alguliev R.M.[1], Aliguliyev R.M.[2], Hajirahimova M.S.[3]
Institute of Information Technology of ANAS, Baku, Azerbaijan
[1]rasim@science.az; [2]a.ramiz@science.az; [3]makrufa@science.az

QUADRATIC BOOLEAN PROGRAMMING MODEL AND BINARY DIFFERENTIAL EVOLUTION ALGORITHM FOR TEXT SUMMARIZATION

We model multi-document summarization as a Quadratic Boolean Programing (QBP) problem where the objective function is a weighted combination of the content coverage and redundancy objectives. The objective function measures the possible summaries based on the identified salient sentences and overlap information between selected sentences. An innovative aspect of our model lies in its ability to remove redundancy while selecting representative sentences. The QBP problem has been solved by using a binary differential evolution algorithm.

***Key words:** text summarization, maximum coverage, less redundancy, optimization model, differential evolution algorithm.*

1. Introduction

Interest in text summarization started with advent of on-line publishing and the increased impact of the Internet and electronic government (e-government) services [1]. With the growing popularity of the Internet and e-government services (for example, electronic document management systems) a huge amount of electronic documents are available online. The increasing amount of electronic documents has led to information overload. In this case, the user due to the large amount of information does not read many relevant and interesting documents. Text summarization is an issue to attack the information overload problem. Text summarization refers to the process of taking a textual document, extracting content from it, and presenting the most important content to the user in a condensed form and in a manner sensitive to the user's or application needs. To achieve this goal, text summarization systems should identify the most salient information in a document and convey it in less space than the original document. Therefore, the text summarization has been used as the useful tools in order to help users efficiently find useful information from immense amount of information [2–4]. Text summarization helps to simplify information search and reduce the search time by pointing the most relevant information that allows users to quickly comprehend the information in a large document [5, 3, 6].

Automatically generating summaries from large text corpora has long been studied in both information retrieval and natural language processing (NLP). There are several types of text summarization tasks. Document summaries can be classified into different types according to different dimensions. For example, extractive summarization can be either generic or query-relevant. Generic document summarization should reflect the major content of the documents without any additional information and prior knowledge. Query-oriented document summarization should focus on the information expressed in the given queries, i.e. the summaries must be biased to the given queries [3, 5–7].

Depending on the number of documents to be summarized, the summary can be a single-document or a multi-document [6]. Single-document summarization can only condense one document into a shorter representation, whereas multi-document summarization can condense a set of documents into a summary. Multi-document 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. It differs from single document summarization in that it is important to identify differences and similarities across documents [3–5].

Automatic document summarization methods can be divided into two categories: supervised and unsupervised methods. Supervised methods are based on algorithms that use a large amount of human-made summaries, and as a result, are most useful for documents that are relevant to the summarizer model. Thus, they do not necessarily produce a satisfactory summary for documents that are not similar to the model. In addition, when users change the purpose of summarization or the characteristics of documents, it becomes necessary to reconstruct the training data or retrain the model. Unsupervised methods do not require training data such as human-made summaries to train the summarizer [3, 5].

There are two types of summarization: extractive summarization and abstractive summarization. Extractive summarization selects the important sentences from the original documents to form a summary, while abstractive summarization paraphrases the corpus using novel sentences. Extractive summarization usually ranks the sentences in the documents according to their scores calculated by a set of predefined features, such as term frequency-inverse sentence frequency (TF-ISF), sentence or term position, and number of keywords. Abstractive summarization usually involves information fusion, sentence compression and reformulation. Although an abstractive summary could be more concise, it requires deep NLP techniques [4, 5, 8, 9]. Therefore, extractive summaries are more feasible and practical. In this paper, we focus on extractive multi-document summarization.

Extractive document summarization clearly entails selecting the most salient information and putting it together in a coherent summary. The summary consists of multiple separately extracted sentences from different documents. Obviously, each of the selected sentences should individually be important. However, when many of the competing sentences are included in the summary, the issue of information overlap between parts of the output comes up, and a mechanism for addressing redundancy is needed [10]. Therefore, when many of the competing sentences are available, given summary length limit, the strategy of selecting best summary rather than selecting best sentences becomes evidently important. Selecting the best summary is a global optimization problem in comparison with the procedure of selecting the best sentences [11]. For content selection, document summarization includes how to identify the important content, remove the redundant content and keep the high content coverage. For linguistic quality, how to keep the content to be coherent and fluent is very significant [12]. In addition, it is known that coverage and redundancy two main criteria that decide the quality of summary. In this paper, we propose a new multi-document summarization approach, called MCLR (Maximum Coverage and Less Redundancy), via sentence extraction to simultaneously deal with these two concerns during sentence selection. We model multi-document summarization as a Quadratic Boolean Programming (QBP) problem where objective function is a weighted combination of the content coverage and redundancy objectives. This model does not only pick sentences from collection with highest significant and takes into account overlap information between selected sentences. The model employs two levels of analysis: first level, every sentence is scored according to the features it covers and second level, when, before being added to the final summary, the sentences deemed to be important are compared to each other and only those that are not too similar to other candidates are included in the final summary. We create a modified differential evolution algorithm to solve the optimization problem.

The rest of paper is organized as follows. In Section 2, we explain the proposed optimization model for multi-document summarization. Next, in Section 3, we give details of binary differential evolution algorithm which has been used to solve the optimization problem. Section 4 addresses the conclusions and future work.

2. Modeling document summarization as an optimization problem

Given a corpus $\mathbf{D}=\{d_1,d_2,...,d_N\}$ of topic-related documents, where N is the number of documents. We represent the corpus as the set of sentences $\mathbf{S}=\{s_1,s_2,...,s_n\}$ from all the documents in the corpus $\mathbf{D}$, where s_i denotes i th sentence in $\mathbf{S}$, n is the number of sentences in the document corpus.

2.1. Sentence representation and similarity measure

In text mining, a textual unit is represented by the weights of the words that it contains, ignoring the order of the words and any punctuation. Formally, in a collection, a sentence is represented by a vector that is defined as a bag-of-words $s_i = \{\omega_{i1}, \omega_{i2}, ..., \omega_{im}\} \in R^m$, where ω_{ik} is the weight of the k th word in the i th sentence and m is the number of words actually appearing in the collection.

Here, the weight ω_{ik} associated with k th word in the sentence s_i is calculated using the tf-isf (term frequency–inverse sentence frequency) scheme:

$$\omega_{ik} = tf_{ik} \times isf_k \, , \tag{1}$$

where tf_{ik} is the number of occurrences of the k th word in sentence s_i, and $isf_k = \log(n/n_k)$, n_k is the number of sentences containing the k th word.

Text similarity measures play an important role in text-related research and applications, in particular, in areas such as text summarization [13], document clustering [14], textual knowledge representation and knowledge discovery [15], and information retrieval [16]. Existing methods for computing text similarity have focused mainly on large documents. We focus on computing the similarity between two sentences. Recently, with the development of NLP applications, the need for an effective and accurate method to compute the similarity between two short or sentence-length text snippets has been identified [15, 17]. Similarity of short text snippets has applications in various computational areas. For example, in text mining short text similarity can be applied as a measure to discover knowledge from textual databases.

In this study, the similarity between two sentences s_i and s_j is defined by the following formula [17]:

$$sim(s_i, s_j) = \sum_{l=1}^{m}\left[\left(\sum_{k=1}^{m}\omega_{ik}p_{kl}\right)\left(\sum_{k=1}^{m}\omega_{jk}p_{kl}\right)\right], \tag{2}$$

where p_{kl} is the similarity between k th and l th words [18]:

$$p_{kl} = \exp(-\mathrm{NGD}_{kl}) \tag{3}$$

In Eq. (3), NGD_{kl} is the Normalized Google Distance between k th and l th words [13, 18]:

$$\mathrm{NGD}_{kl} = \frac{\max\{\log(n_k), \log(n_l)\} - \log(n_{kl})}{\log n - \min\{\log(n_k), \log(n_l)\}}, \tag{4}$$

where n_{kl} denotes the number of sentences in the collection containing both k th and l th words.

2.2. Optimization model

Most of the state-of-art summarization systems are based on extracting the most salient and non-redundant sentences to composite the final summaries. Upon this extractive summarization framework, sentences first evaluated and ranked according to certain criteria and measures, and then the most significant ones are extracted from the original documents to generate a summary automatically. Without doubt, each of the selected sentences included in the summary should be individually important. However, this does not guarantee they collectively produce the best summary. For example, if the selected sentences overlap a lot with each other, such a summary is definitely not desired.

Document summarization, especially multi-document summarization in essence is a multi-objective optimization problem. It requires the simultaneous optimization of more than one objective function. A particular challenge for multi-document summarization is that a document set might contain diverse information either related or unrelated to the main topic. Hence, we need effective summarization methods to analyze the information stored in different documents and extract the globally important information to reflect the main topic. Another challenge for multi-document summarization is that the information stored in different documents inevitably overlaps with each other, and hence we need effective summarization methods to merge information stored in different documents, and if possible, contrast their differences [11]. In this study, when building summaries from multiple documents, our approach generally attempts to optimize two objectives:

- *Content coverage:* The content coverage means that the generated summary should cover all subtopics as much as possible. It concerns the extent to which the information provided in original documents is included in the generated summary. A good summary should maximize this goal to its best.
- *Redundancy*: It is expected that the redundant or the duplicate information contained in the generated summary be minimized.

Optimizing these properties jointly is a challenging task. This is because the inclusion of relevant sentences relies on not only properties of the sentences themselves, but also properties of every other sentence in the summary. Unlike single-document summarization, redundancy is particularly important since it is likely that sentences from different documents will convey the same information [11].

We define two objective functions:

1) $f_{\text{cov}}(s_i)$: The content coverage of sentence s_i participating in the summary:

$$f_{\text{cov}}(s_i) = sim(s_i, O), \quad i = 1,..n, \tag{5}$$

where O is the center of the collection $\mathbf{S} = \{s_1, s_2, ..., s_n\}$.

It is known that [19] the centre O reflects the main content of collection $\mathbf{S} = \{s_1, s_2, ..., s_n\}$. Thus, Eq.(5) evaluates the importance of sentence s_i by measuring its similarity to the centre O. k th coordinate o_k of the centre $O = [o_1, o_2, ..., o_m]$ is calculated as: $o_k = \frac{1}{n}\sum_{i=1}^{n} w_{ij}$, $k = 1,...,m$. Higher value of $f_{\text{cov}}(s_i)$ corresponds to higher content coverage of sentence s_i.

2) $f_{\text{red}}(s_i, s_j)$: The redundancy between sentences s_i and s_j:

$$f_{\text{red}}(s_i, s_j) = 1 - sim(s_i, s_j), \quad i \neq j = 1,..n. \tag{6}$$

Higher value of $f_{\text{red}}(s_i, s_j)$ corresponds to lower overlap in content between sentences s_i and s_j, i.e. higher value of objective (6) provides minimum redundancy (high diversity) in the summary.

Let x_i be binary variable, $x_i = 1$ when s_i is selected; otherwise, $x_i = 0$. Since, a high quality summary should maximize the content coverage of the given document set, while minimize the redundancy, then text summarization problem can then be formalized as the following optimization problem:

maximize

$$f(\mathrm{X}) = w \cdot f_{\text{cov}}(\mathrm{X}) + (1-w) \cdot f_{\text{red}}(\mathrm{X}) = w \cdot \sum_{i=1}^{n-1} sim(s_i, O)x_i + (1-w) \cdot \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(1 - sim(s_i, s_j))\, x_i x_j \tag{7}$$

subject to

$$\sum_{i=1}^{n} l_i\, x_i \leq L, \tag{8}$$

$$x_i \in \{0,1\}, \ \forall i, \tag{9}$$

where L is the given summary length limitation, l_i indicates the length of sentence s_i. The number of words or in bytes measures the lengths of summary and sentence.

The optimization model (7)-(9) balances content coverage and diversity of the summary. In this model a parameter w is used to combine the two objectives (5) and (6) into a scalar (7). w is the weighting parameter, specifying the relative contributions of the $f_{\text{cov}}(\cdot)$ and $f_{\text{red}}(\cdot)$ functions to the hybrid function $f(\cdot)$. When $w=1$, this model gives preference to sentences that are maximally relevant to the content of the document regardless of the diversity. In contrast, when $w=0$, the model gives preference to sentences that diverge from the others regardless of the content coverage. By varying the parameter value w and solving a sequence of quadratic Boolean programming (QBP) problems (7)–(9) (for each w) the efficient summaries from the maximum content coverage summary (w=1) to the high diversity summary (w=0) can be found. If $w=0.5$ the $f_{\text{cov}}(\cdot)$ and $f_{\text{red}}(\cdot)$ functions are assumed to be equally important. In our study we set $w=0.75$.

Now our objective is to find the binary assignment $\mathrm{X}=[x_i]$ with the best content coverage and least redundancy such that the summary length is at most L. The basic step of multi-document summarization is to extract a candidate sentence. If the length L is the number of terms in the summary, the cost of each candidate sentence is the number of terms within it. The cardinality constraint (8) guarantees that the summary is bounded in length. The integrality constraint on x_i (9) is automatically satisfied in the problem above.

3. Binary differential evolution algorithm

In solving optimization problems with a high-dimensional search space, the classical optimization algorithms do not provide a suitable solution because the search space increases exponentially with the problem size, therefore solving these problems using exact techniques is not practical. Over the last decades, there has been a growing interest in algorithms inspired by the behaviors of natural phenomena. It is shown by many researchers that these algorithms are well suited to solve complex computational problems such as optimization of objective functions [20, 21], pattern recognition [22, 23], document clustering [14], and text summarization [13, 18].

In our study, the optimization problem (7)-(9) was solved using a differential evolution (DE) [18, 21–22]. The execution of the DE is similar to other evolutionary algorithms like genetic algorithms or evolution strategies.

3.1. Population initialization

The evolutionary algorithms differ mainly in the representation of parameters and in the evolutionary operators. The classical DE is a population-based global optimization that uses a real-coded representation. Like to other evolutionary algorithms, DE also starts with a population of N_{pop} individuals $\mathbf{P}=[X_1, X_2, ..., X_{N_{pop}}]$, where individual $X_p=[x_{p,1}, x_{p,2}, ..., x_{p,n}]$ ($p=1,2,...,N_{pop}$) is an n-dimensional vector with parameter values determined randomly and uniformly between predefined search ranges $[X_{\min}, X_{\max}]$, where $X_{\min}=[x_{\min,1}, x_{\min,2}, ..., x_{\min,n}]$ and $X_{\max}=[x_{\max,1}, x_{\max,2}, ..., x_{\max,n}]$. Then mutation and crossover operators are employed to generate new candidate vectors, and a selection scheme is applied to determine whether the offspring or the parent survives to the next generation. The above process is repeated until a termination criterion is reached.

3.2. Mutation

A mutant vector, denoted as $Y_p(t)=[y_{p,1}(t), y_{p,2}(t), ..., y_{p,n}(t)]$ ($p=1,2,...,N_{pop}$) is generated by using a mutation operator. The different mutation strategies are developed in literature [18, 21-24]. In this study, for each target vector $X_p(t)$ randomly choose two other vectors $X_{p1}(t)$ and $X_{p2}(t)$ from the same generation. Then it calculates the weighting combination of the differences $(X_p(t)-X_{p1}(t))$, $(X_p(t)-X_{p2}(t))$ and creates a mutant vector $Y_p(t)$ by adding the result to the best current solution $X^{best}(t)$. Thus, for the i th component of the mutant vector $Y_p(t)$ we obtain:

$$y_{p,i}(t) = x_i^{best}(t) + (\lambda_p - \lambda_{p1})(x_{p,i}(t) - x_{p1,i}(t)) + (\lambda_p - \lambda_{p2})(x_{p,i}(t) - x_{p2,i}(t)) \ . \tag{10}$$

The coefficients λ_p, λ_{p1}, and λ_{p2} we define as follows:

$$\lambda_p = \frac{|f(X_p)|}{|f(X_p)| + |f(X_{p1})| + |f(X_{p2})|}, \ \lambda_{p1} = \frac{|f(X_{p1})|}{|f(X_p)| + |f(X_{p1})| + |f(X_{p2})|}, \ \lambda_{p2} = \frac{|f(X_{p2})|}{|f(X_p)| + |f(X_{p1})| + |f(X_{p2})|}, \tag{11}$$

where $f(\cdot)$ is the objective function (7).

3.3. Crossover

After the mutation phase, a crossover operator is applied to each mutant vector and its corresponding target vector to yield a trial vector. A crossover operation comes into play after generating the mutant vector to enhance the potential diversity of the population. The mutant vector $Y_p(t)$ exchanges its components with the target vector $X_a(t)$ under this operation to form the trial vector $Z_p(t) = [z_{p,1}(t), z_{p,2}(t), ..., z_{p,n}(t)]$. Two commonly used crossover operations are the binomial crossover and the exponential crossover. In our study, the binomial crossover is employed and executed as follows:

$$z_{p,i}(t) = \begin{cases} y_{p,i}(t), \text{ if } \text{rand}_{p,i} \le CR \text{ or } i = i_{rand} \\ x_{p,i}(t), \text{ otherwise} \end{cases}, \tag{12}$$

where the index i_{rand} refers to a randomly chosen integer in the set $\{1,2,...,n\}$ which is used to ensure that at least one component of the trial vector, $Z_p(t)$, differs from its target vector, $X_p(t)$. CR is the real-valued crossover rate in the range $[0,1]$ which is set by the user and remains constant during the search process; $\text{rand}_{p,i}$ is the uniformly distributed random number within the range $(0,1)$ chosen once for each ith component of the pth parameter vector, $i \in \{1,2,...,n\}$, $p \in \{1,2,...,N_{pop}\}$.

3.4. Selection

To keep the population size constant over subsequent generations, the next step of the algorithm calls for selection to determine whether the target or the trial vector survives to the next generation, i.e., at $t+1$. The selection operation is described as:

$$X_p(t+1) = \begin{cases} Z_p(t), \text{ if } f(Z_p(t)) \ge f(X_p(t)) \\ X_p(t), \text{otherwise} \end{cases}. \tag{13}$$

Therefore, if the new trial vector yields an equal or higher value of the objective function, it replaces the corresponding target vector in the next generation; otherwise the target vector 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.

3.5. Discretization

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:

$$x_{p,i}(t+1) = \begin{cases} 1, \text{ if } \text{rand}_{p,i} < sigm(x_{p,i}(t+1)) \\ 0, \text{ otherwise} \end{cases}, \tag{14}$$

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

$$sigm(z) = \frac{1}{1+\exp(-z)}. \tag{15}$$

Using this transformation from the real-coded representation we obtain the binary-coded representation, $x_{p,i}(t) \in \{0,1\}$. Where the $x_{p,i}(t) = 1$ indicates that the i th sentence is selected to be included in the summary, otherwise, the i th sentence is not be selected.

3.6. Termination criterion

Mutation, crossover and selection operations continue until some termination criterion is reached. The termination criterion can be defined in a few ways like: 1) by a fixed number of iterations $t_{\max}$, with a suitably large value of $t_{\max}$ depending upon the complexity of the objective function; 2) when best fitness of the population does not change appreciably over successive iterations; 3) by a specified CPU time limit; and alternatively 4) attaining a pre-specified objective function value [22, 23]. According to our previous successful experience [26, 27], in this paper we use the first one as the termination criteria, i.e., the algorithm terminates when the maximum number of generations $t_{\max}$ is achieved.

3.7. Framework of the binary DE algorithm

Based on the above initialization, mutation, crossover, selection and discretization operations the framework of the binary DE algorithm can be summarized as:

Step 1: *Initialization*. Set the generation number $t=0$, and randomly initialize a population of N_{pop} target vectors, $\mathbf{P} = [X_1(t), X_2(t),\ldots,X_{N_{pop}}(t)]$, with $X_p = [x_{p,1}(t), x_{p,2}(t),\ldots,x_{p,n}(t)]$ uniformly distributed in the range $[X_{\min}, X_{\max}]$, $p = 1,2,\ldots,N_{pop}$.

Step 2: *Discretization*. Transform real-coded vectors to binary-coded vectors using Eq. (14).

Step 3: *Evaluation*. Evaluate each vector in $\mathbf{P} = [X_1, X_2,\ldots,X_{N_{pop}}]$ and select the vector with current best solution.

Step 4: *Mutation*. Generate a mutant vector $Y_p(t) = [y_{p,1}(t), y_{p,2}(t),\ldots,y_{p,n}(t)]$ for target vector $X_p(t)$ by using mutation operator (10), $p = 1,2,\ldots,N_{pop}$.

Step 5: *Crossover*. Generate a trial vector $Z_p(t)$ for target vector $X_p(t)$ by applying crossover operator (12) on $Y_p(t)$ and $X_p(t)$, $p = 1,2,\ldots,N_{pop}$.

Step 6: *Selection*. Evaluate each $Z_p(t)$ and determine the members of the target population of the next generation by using the selection scheme (13), $p = 1,2,\ldots,N_{pop}$.

Step 7: *Discretization*. **Discretizate** a new trial vector $X_p(t+1)$ by using Eq. (14), $p = 1,2,\ldots,N_{pop}$.

Step 8: *Stopping*. Repeat steps 2–7 until a user-specified maximum number $t_{\max}$ of fitness calculation is reached.

Step 9: *Output*. Report the summary obtained by the best vector $X^{best}(t)$ as the final solution at maximum number of iteration.

3.8. Runtime complexity analysis

In this section, we analyze the time complexity of the proposed algorithm. Runtime-complexity analysis of the population-based stochastic search techniques like DE is a critical issue by its own right [9, 21–22,]. Das and Suganthan [21] note that the average runtime of a standard DE algorithm usually depends on the population size, length of the vector and its stopping criterion. The authors pointed out that in each generation of DE a loop over N_{pop} is conducted, containing a loop over *n*.

Assuming N_{pop} and *n* are the population size and the length of each vector in the DE, respectively, the time complexity of one generation for DE can be estimated as follows:

1. Time required for initialization of the individual is proportional to the length of the individual. As the length of the vector is equal to *n*, the time complexity of population initialization is $O(N_{pop} \times n)$.
2. Since the mutation and crossover operations are performed at the component level for each DE vector, the number of fundamental operations in DE is proportional to the total number of loops. Thus, mutation and crossover require $O(N_{pop} \times n)$ time each.
3. The time complexity for selection is $O(N_{pop})$.
4. Fitness computation is composed of three steps:
   - Complexity of computing similarity of *n* sentences to the centre of document collection is $O(N_{pop} \times n \times m)$.
   - For updating the centers (of summaries) total complexity is $O(N_{pop} \times m)$.
   - Complexity of computing similarity between *n* sentences is $O(N_{pop} \times m \times n^2)$.

Therefore, the fitness evaluation has total complexity $O(N_{pop} \times m \times n^2)$.

Thus summing up the above complexities, total time complexity becomes $O(N_{pop} \times m \times n^2)$ per generation. For maximum $t_{\max}$ number of generations total complexity becomes $O(N_{pop} \times m \times n^2 \times t_{\max})$.

4. Conclusion

With the explosive growth of the volume and complexity of document data (e.g., news, blogs, web pages) on the Internet and electronic government multi-document summarization provides a useful solution for understanding documents and reducing information overload. Thus, multi-document summarization has attracted much attention in recent years, and many applications have been developed. Multi-document summarization aims to generate a compressed summary by extracting the major information in a collection of documents sharing the same or similar topics. In multi-document summarization, the risk of extracting two sentences conveying the same information is greater than in a single-document summarization problematic. Moreover, identifying redundancy is a critical task, as information appearing several times in different documents can be qualified as important. Hence, we need effective summarization methods to analyze and extract the important information. A good summary is expected to preserve the topic information contained in the documents as much as possible, and at the same time to contain as little redundancy as possible, known as information richness and diversity, respectively. The requirement raises a fundamental problem: how important will a selected summary be to represent the whole documents?

This paper discusses work on multi-document summarization to create a generic extractive summary of multiple documents on the same (or related) topic. The proposed approach adopts a broadly used summarization model – sentence extraction – to extract important sentences and compose them into a summary. This approach divides the multi-document summarization task into three subtasks: (1) evaluating sentences according to their importance of being part in the summary by calculating their similarity to the centre of sentences collection, (2) eliminating redundancy while extracting the most important sentences, and (3) organizing extracted sentences into a summary.

We present a multi-document summarization model which extracts key sentences from given documents while reducing redundant information in the summaries. The model is represented as a QBP problem that was solved by using a binary differential evolution algorithm.

References

1. **Hung S.Y., Tang K.Z., Chang C.M., Ke C.D.** User acceptance of intergovernmental services: an example of electronic document management system // Government Information Quarterly, 2009, vol.26, №2, pp. 387–397.
2. Ko Y. and Seo J. An effective sentence-extraction technique using contextual information and statistical approaches for text summarization // Pattern Recognition Letters, 2008, vol.29, №9, pp.1366–1371.
3. Mani I. and Maybury M.T. Advances in automatic text summarization, MIT Press, Cambridge, 1999, 442 pp.
4. Ouyang Y., Li W., Li S., Lu Q. Applying regression models to query-focused multi-document summarization // Information Processing & Management, 2011, vol.47, №2, pp.227–237.
5. Kutlu M., Cigir C., Cicekli I. Generic text summarization for Turkish // The Computer Journal, 2010, vol.53, №8, pp.1315-1323.
6. Wan X., Xiao J. Exploiting neighborhood knowledge for single document summarization and keyphrase extraction // ACM Transactions on Information Systems, 2010, vol.28, №2, 8:1–8:34.
7. Tang J., Yao L., Chen D. Multi-topic based query-oriented summarization / Proceedings of the 9th SIAM International Conference on Data Mining, Nevada, USA, 2009, april 30 may 2, pp.1148–1159.
8. McDonald R. A study of global inference algorithms in multi-document summarization / Proceedings of the 29th European Conference on IR Research, Rome, Italy, Springer-Verlag, LNCS, 2007, April 2–5, №25, pp. 557–564.
9. Wang Y., Li B., Weise T. Estimation of distribution and differential evolution cooperation for large scale economic load dispatch optimization of power systems // Information Sciences, 2010, vol.180, №12, pp.2405–2420.
10. Chali Y., Hasan S.A., Joty S.R. Improving graph-based random walks for complex question answering using syntactic, shallow semantic and extended string subsequence kernels // Information Processing & Management, 2011, vol.47, №6, pp. 843-855.
11. Huang L., He Y., Wei F., Li W. Modeling document summarization as multi-objective optimization / Proceedings of the Third International Symposium on Intelligent Information Technology and Security Informatics, Jinggangshan, China, 2010, april 02–04, pp.382–386.
12. He R., Qin B., Liu T. A novel approach to update summarization using evolutionary manifold-ranking and spectral clustering // Expert Systems with Applications, 2012, vol.39, №3, pp.2375–2384.
13. Aliguliyev R.M. A new sentence similarity measure and sentence based extractive technique for automatic text summarization // Expert Systems with Applications, 2009, vol.36, №4, pp.7764–7772.
14. Aliguliyev R.M. Performance evaluation of density-based clustering methods // Information Sciences, 2009, vol.179, №20, pp.3583–3602.
15. Islam A., Inkpen D. Semantic text similarity using corpus-based word similarity and string similarity // ACM Transactions on Knowledge Discovery from Data, 2008, vol.2, №2, pp.10:1-10:25.
16. Tsai F.S., Tang W., Chan K.L. Evaluation of novelty metrics for sentence-level novelty mining // Information Sciences, 2010, vol.180, №12, pp.2359–2374.
17. Wenyin L., Quan X., Feng M., Qiu B. A short text modeling method combining semantic and statistical information // Information Sciences, 2010, vol.180, no.20, pp.4031–4041.
18. Alguliev R.M., Aliguliyev R.M. Evolutionary algorithm for extractive text summarization //Intelligent Information Management, 2009, vol.1, №2, pp.128–138.
19. Radev D., Jing H., Stys M., Tam D. Centroid-based summarization of multiple documents // nformation Processing & Management, 2004, vol.40, №6, pp.919–938.

20. Rashedi E., Nezamabadi-pour H., Saryazdi S. GSA: a gravitational search algorithm, // Information Sciences, 2009, vol.179, №13, pp.2232–2248.
21. Zielinski K., Peters D., Laur R. Runtime analysis regarding stopping criteria for differential evolution and particle swarm optimization / Proceedings of the 1st International Conference on Experiments/Process/System Modelling/Simulation /Optimization, Athens, Greece, 2005, july 6–9.
22. Das S., Suganthan P.N. Differential evolution: a survey of the state-of-the-art, // IEEE Transactions on Evolutionary Computation, 2011, vol.15, №1, pp.4–31.
23. Das S., Sil S. Kernel-induced fuzzy clustering of image pixels with an improved differential evolution algorithm // Information Sciences, 2010, vol.180, №8, pp.1237–1256.
24. Lu Y., Zhou J., Qin H., Li Y., Zhang Y. An adaptive hybrid differential evolution algorithm for dynamic economic dispatch with valve-point effects // Expert Systems with Applications, 2010, vol.37, №7, pp.4842–4849.
25. Zhang M., Luo W., Wang X. Differential evolution with dynamic stochastic selection for constrained optimization // Information Sciences, 2008, vol.178, №15, pp.3043–3074.
26. Alguliev R.M., Aliguliyev R.M., Hajirahimova M.S., Mehdiyev C.A. MCMR: maximum coverage and minimum redundant text summarization model // Expert Systems with Applications, 2011, vol.38, №12, pp.14514–14522.
27. Aliguliyev R.M. Clustering techniques and discrete particle swarm optimization algorithm for multi-document summarization // Computational Intelligence, 2010, vol.26, №4, pp.420–448.

UOT 004.02

Əliquliyev Rasim M.[1], Alıquliyev Ramiz M.[2], Hacırəhimova Məkrufə Ş.[3]
AMEA İnformasiya Texnologiyaları İnstitutu, Bakı, Azərbaycan
[1]rasim@science.az; [2]a.ramiz@science.az; [3]makrufa@science.az

Mətnlərin referatlaşdırılması üçün kvadratik Bul proqramlaşdırma modeli və binar diferensial təkamül alqoritmi

Məqalədə mətnlərin avtomatik referatlaşdırılması kvadratik Bul proqramlaşdırma məsələsi kimi formalizə edilmişdir. Təklif olunan model mətnlərin optimal referatının yaradılmasına imkan verir. Optimallaşma məsələsinin həlli üçün binar diferensial təkamül alqoritmi işlənmişdir.

Açar sözlər: *mətn referatlaşdırması, maksimum əhatəli, kiçik qalıq, optimallaşma modeli, diferensial təkamül alqoritmi.*

УДК 004.02

Алгулиев Расим М.[1], Алыгулиев Рамиз М.[2], Гаджирагимова Макруфа Ш.[3]
Институт Информационных Технологии НАНА, Баку, Азербайджан
[1]rasim@science.az; [2]a.ramiz@science.az; [3]makrufa@science.az

Модель квадратичного булевого программирования и бинарный дифференциальный эволюционный алгоритм для реферирования текстов

В статье проблема автоматического реферирования текстов сформулирована как задача квадратичного программирования с булевыми переменными. Предложенная модель позволяет формировать оптимальный реферат текстов. Для решения задачи оптимизации разработан бинарный алгоритм дифференциальной эволюции.

Ключевые слова: *реферирование текстов, максимальный охват, наименьшая избыточность, оптимизационная модель, алгоритм дифференциальной эволюции.*