Extractive Single-Document Summarization Based on Global-Best Harmony Search and a Greedy Local Optimizer

Martha Mendoza[1(✉)], Carlos Cobos[1], and Elizabeth León[2]

[1] Universidad Del Cauca, Popayán, Colombia
{mmendoza, ccobos}@unicauca.edu.co
[2] Universidad Nacional de Colombia, Bogotá D.C., Colombia
eleonguz@unal.edu.co

Abstract. Due to the great amount of documents available on the Web, end users need to be able to access information in summary form – keeping the most important information in the document. The methods employed for automatic text summarization generally allocate a score to each sentence in the document, taking into account certain features. The most relevant sentences are then selected, according to the score obtained for each sentence. In this paper, the extractive single document summarization task is treated as a binary optimization problem and, based on the Global-best Harmony Search metaheuristic and a greedy local search procedure, a new algorithm called ESDS-GHS-GLO is proposed. This algorithm optimizes an objective function, which is a lineal normalized combination of the position of the sentence in the document, sentence length, and coverage of the selected sentences in the summary. The proposed method was compared with the state of the art methods MA-SingleDocSum, DE, FEOM, UnifiedRank, Net-Sum, QCS, CRF, SVM, and Manifold Ranking, using ROUGE measures on the DUC2001 and DUC2002 datasets. The results showed that ESDS-GHS-GLO outperforms most of the state-of-the-art methods except MA-SingleDocSum. ESDS-GHS-GLO obtains promissory results using a fitness function less complex than MA-SingleDocSum, therefore requiring less execution time.

Keywords: Single-document summarization · Memetic algorithms · Global-best harmony search · Greedy search

1 Introduction

Today, automatic text summarization constitutes a key service for a range of application types, including internet, library, scientific, and business uses [1]. The vast quantities of information stored in digital text documents need summaries in order to help users find the required information with the least time and effort possible. For many years, the automatic generation of summaries has attempted to create summaries that closely approximate those generated by humans [1, 2], but until now, this research area is still unresolved.

Different taxonomies for summaries exist [1, 2] based on the way the summary is generated, the target audience of the summary, the number of documents to be

O. Pichardo Lagunas et al. (Eds.): MICAI 2015, Part II, LNAI 9414, pp. 52–66, 2015.
DOI: 10.1007/978-3-319-27101-9_4

summarized, and so on. According to how it is generated, the summary can be either extractive or abstractive [1, 2]. Extractive summaries are formed from the reuse of portions of the original text. Abstractive summaries [3] on the other hand are rather more complex, requiring linguistic analysis tools to construct new sentences from those previously extracted. Taking account of the target audience, summaries may be [1, 2] generic, query-based, user-focused or topic-focused. Generic summaries do not depend on the audience for whom the summary is intended. Query-based summaries respond to a query made by the user. User-focused ones generate summaries to tailor the interests of a particular user, while topic-focused summaries emphasize those summaries on specific topics of documents. Depending on the number of documents processed, summaries [1, 2] can be either single document or multiple document. As regards the language of the document, they may be monolingual or multilingual, and regarding document genre may be scientific article, news, blogs, and so on. The summarization algorithm (method) proposed in this paper is extractive, for a single document, and for any type of document, although the evaluation was performed on a set of news.

Automatic summarization is an area that has explored different methods for the automatic generation of single document summaries, such as (1) statistical and probabilistic approaches, which use information such as the frequency of occurrence of a term in a text, the position of the sentences in the document, and the presence of keywords or words from the document title in the sentences [4]; (2) Machine learning approaches, including Bayes' Theorem [5, 6], Hidden Markov Models [7, 8], Neural networks [9], Conditional Random Fields [10], Probabilistic Support Vector Machine (PSVM) and Naïve Bayes [11]; (3) Text connectivity approaches [12, 13], including lexical chains [14] and rhetorical structure theory [15]; (4) Graph-based approaches [16, 17], which represent sentences in the vertices of the graph and the similarity between the text units by means of the edges, then an iterative process is carried out and the summary with sentences from the first vertices is obtained; (5) Algebraic approaches using Latent Semantic Analysis [18] based on Singular Value Decomposition [19–21] or Non-negative Matrix Factorization [22]; (6) Metaheuristic approaches that seek to optimize an objective function to find the sentences that will be part of the summary. These works include genetic algorithms, [23–28], particle swarm optimization (PSO) [29], Harmony Search [30], and Differential Evolution (DE) algorithm [31, 32]; and (7) Fuzzy approaches that combine fuzzy set theory with swarm intelligence (binary PSO) [33] or with clustering and evolutionary algorithms in a new fuzzy evolutionary optimization model (FEOM) [34] for document summarization.

Algebraic, clustering, probabilistic, metaheuristic and fuzzy approaches are language independent and unsupervised, two key aspects on which more emphasis is being placed in the most recent research. Research based on a memetic algorithm (combination of metaheuristics) for single document summarization [35] has recently shown good results, making this a promising area. Therefore, in this paper, a new memetic algorithm for the automatic generation of extractive and generic single document summaries is proposed.

The new memetic algorithm is based on Global-best Harmony Search (GHS) bearing in mind that "No Free Lunch theorems for optimization state that no one algorithm is better than any other when performances are averaged over the whole set of possible problems. However, it has been recently suggested that algorithms might show performance differences when a set of real-world problems is under study" [36] and that GHS [37] is showing promissory results in a great variety of real problems (continuos, discrete, and binary problems) [38]. The memetic algorithm also includes a greedy search as local search operator. The new algorithm, ESDS-GHS-GLO optimizes an objective function expressed by the lineal and normalized combination of three factors: position of the sentences selected in the candidate summary; length of sentences selected in the candidate summary; and coverage of the candidate summary, i.e. cosine similarity between all candidate sentences in the summary and a global representation of the document.

The rest of the paper is organized as follows: Sect. 2 introduces document representation and characteristics of the objective function proposed in the algorithm. Section 3 describes the proposed algorithm; while the results of evaluation using data sets, along with a comparison and analysis with other state-of-the-art methods, are presented in Sect. 4; finally, Sect. 5 presents conclusions and future work.

2 Problem Statement and Its Mathematical Formulation

The representation of a document is made based on the vector space model proposed by Salton [39]. Thus, a document is represented by the sentences that compose it, i.e. $D = \{S_1, S_2, \ldots, S_n\}$, where S_i corresponds to the i-th sentence of the document and n is the total number of sentences in the document. Likewise, a sentence is represented by the set $S_i = \{t_{i,1}, t_{i,2}, \ldots, t_{i,j}, \ldots, t_{i,o}\}$, where $t_{i,j}$ is the j-th term of the sentence S_i and o is the total number of terms in the sentence. Thus, the vector representation of a sentence of the document is a vector containing the weights of the terms, as shown in Eq. (1)

$$S_i = \{w_{i,1}, w_{i,2}, \ldots, w_{i,k}, \ldots, w_{i,m}\} \tag{1}$$

where m is the number of distinct terms in the document collection and $w_{i,k}$ is the weight of the k-th term in sentence S_i.

The component $w_{i,k}$ is calculated using the Okapi BM25 formula [39] (see Eq. (2))

$$w_{i,k} = \frac{(k_1 + 1) \times f_{i,k}}{k_1 \times \left((1 - B) + B \times \left(\frac{L_i}{L_{AVG}}\right)\right) + f_{i,k}} \times log\left(\frac{n}{n_k}\right) \tag{2}$$

where $f_{i,k}$ represents the frequency of the k-th term in sentence S_i, L_i is the length of sentence S_i, L_{AVG} is the average of all sentences in the document, n_k denotes the number of sentences in which the k-th term appears, and n is the number of sentences in the document collection. k_1 and B are two tuning parameters equal to 2 and 0.75 respectively.

Thus the aim of generating a summary of a single document is to obtain a subset of D with the sentences that contain the main information of the document. To do this, characteristics are used whose purpose is to evaluate the subset of sentences to determine the extent to which they cover the most relevant information of the document. One of these characteristics (coverage) is based on measures of similarity between sentences. The similarity between two sentences S_i and S_j, according to the vector representation described, is measured in the same way as the cosine similarity [39], which relates to the angle of the vectors S_i and S_j.

In the proposed algorithm, the objective function is in charge of guiding the search for the best summaries based on sentence characteristics. In this paper, an objective function based on the lineal normalized combination of sentence position, sentence length, and coverage of the selected sentences is proposed [40, 41].

Position Factor. According to previous studies, the relevant information in a document, regardless of its domain [42], tends to be found in certain sections such as titles, headings, the leading sentences of paragraphs, the opening paragraphs, etc. In this research, the position factor (PF) is calculated using Eq. (3)

$$\begin{aligned} PF_s &= \frac{APF_s - \min_{\forall Summary} PF}{\max_{\forall Summary} PF - \min_{\forall Summary} PF} \\ APF_s &= \sum_{\forall S_i \in Summary} \frac{PositionRanking(S_i)}{O} \end{aligned} \tag{3}$$

where APF_s is the average sentence position in the summary S, O is the number of sentences in the summary S, $max_{\forall summary}\, PF$ is the average of the maximum O values obtained from the position rankings of all sentences in the document (i.e. the average top maximum O position rankings of all sentences), $min_{\forall summary}\, PF$ is the average of the minimum O values obtained from the position rankings of all sentences in the document, and PF_s is the position factor of the sentences of the summary S. *PositionRanking*(S_i) is the position ranking of sentence S_i calculated by Eq. (4)

$$PositionRanking(S_i) = \frac{2 - 2 * \left(\frac{i-1}{n-1}\right)}{n} \tag{4}$$

where i is the position of the sentence in order of occurrence in the document, and n is the total number of sentences in the document. This formula is based on that used in the linear-rank selection method in genetic algorithms. The best ranking receives a value of *2/n* and the lowest ranking is close to zero but not zero.

PF_s is close to one (1) when sentences in the summary are the first sentences in the document and PF_s is close to zero (0) when sentences in the summary are the last in the document. The *max* and *min* components in PF_s allow the normalization of the factor between zero and one (Min-Max normalization commonly used in data mining and other areas).

Length Factor. Some studies have concluded that the shortest sentences of a document ought to be less likely to appear in the document summary [6]. Equation (5) shows the calculation of length factor for the sentences of a summary:

$$\begin{aligned} LF_s &= \frac{ALF_s - \min_{\forall Summary} LF}{\max_{\forall Summary} LF - \min_{\forall Summary} LF} \\ ALF_s &= \sum_{\forall S_i \in Summary} \frac{Length(S_i)}{O} \end{aligned} \tag{5}$$

where ALF_s is the average sentence length in the summary S, $Length(S_i)$ is the length (in words) of sentence S_i, O is the number of sentences in the summary S, $max_{\forall summary}$ LF is the average of the maximum O values obtained from the lengths of all sentences in the document (i.e. the average top maximum O lengths of all sentences), $min_{\forall summary}$ LF is the average of the minimum O values obtained from the lengths of all sentences in the document, and LF_s is the length factor of the sentences of the summary S. LF_s is close to one (1) when sentences in the summary are the largest sentences in the document and LF_s is close to zero (0) when sentences in the summary are the shortest in the document. The *min* and *max* components in LF_s allow the normalization of the factor between zero and one.

Coverage Factor. A summary ought to contain the main aspects of the documents with the least loss of information. The sentences selected should therefore cover the largest amount of information contained within the set of sentences in the document. As such, coverage factor is calculated taking into account the cosine similarity between the text of the candidate summary and all sentences of the document, as shown in Eq. (6).

$$CF_s = sim_{cos}(R, D) \tag{6}$$

where R represents the text with all the candidate summary sentences; D represents all the sentences of the document collection (in this case, it is the centroid of the document). This factor therefore takes values between zero and one, but bearing in mind that length summary is just a portion θ of the entire document, the real range of this factor is between θ-ε and $\theta + \varepsilon$, where θ-$\varepsilon > 0$ and $\theta + \varepsilon \ll 1$. NB: in order to compare this factor with position and length factors, all values for candidate summaries in the iterative process should be normalized based on a Min-Max strategy using current solution values in the optimization algorithm.

Thus the objective function to be maximized is defined as the linear normalized combination of sentence position (PF_s), sentence length (LF_s), and coverage (CF_s) factors (see Eq. (7)). Alfa (α), Beta (β), and Gamma (γ) coefficients are introduced, which gives flexibility to the objective function allowing more or less weight to be given to each factor. The sum of these coefficients should be equal to one, i.e. $\alpha + \beta + \gamma = 1$. Equation (8) includes a restriction to maximize the information included in the summary by selecting sentences containing relevant information but few words.

$$\text{Maximize}\, f(x) = \alpha * PF_s + \beta * LF_s + \gamma * CF_s \tag{7}$$

$$\text{subject to } \sum\nolimits_{i=1}^{r} l_i x_i \leq L \tag{8}$$

where x_i indicates one if the sentence S_i is selected and zero otherwise; l_i is the length of the sentence S_i (measured in words) and L is the maximum number of words allowed in the generated summary.

3 The Proposed Memetic Algorithm

Global-best Harmony Search (GHS) is a stochastic optimization algorithm proposed in 2008 by Mahamed G.H. Omran and Mehrdad Mahdavi [37], which hybridizes the original Harmony Search (HS) with the concept of swarm intelligence proposed in PSO (Particle Swarm Optimization) [37], in which a swarm of individuals (called particles) fly through the search space. Each particle represents a candidate solution to the optimization problem. The position of a particle is influenced by the best position visited by itself (own experience) and the position of the best particles in the swarm (swarm experience). GHS modifies the pitch adjustment step in the original HS in such a way that the newly-produced harmony can mimic the best one in the harmony memory. This allows GHS to work efficiently in continuous and discrete problems. GHS is generally better than the original HS when applied to problems of high dimensionality and when noise is present [37].

In Fig. 1, the general outline of ESDS-GHS-GLO, the proposed memetic algorithm for automatically generating extractive summaries based on Global-best Harmony Search [37] and greedy search, is shown.

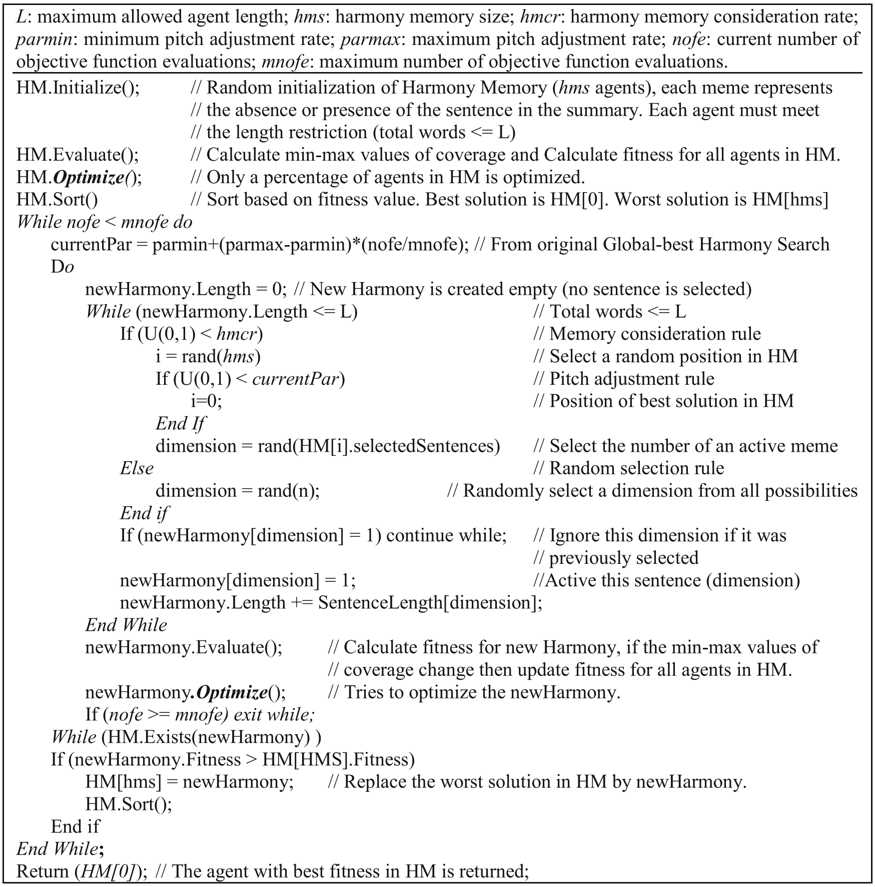
Harmony Memory Initialization (HM.Initialize). The initial population is composed of p agents, generated randomly, taking into account the constraint of the maximum number of words allowed in the summary (the number of sentences in the agent is controlled by means of Eq. (8)). Each agent represents the presence of the sentence in the summary with a one, absence with a zero. The most common strategy for initializing the population ($t = 0$) is to randomly generate each agent. In order that all the sentences in the document have the same probability of being part of the agent, a random number between one and n (number of sentences in the document) is defined, the gene corresponding to this value is chosen and a value of one is given, so that this sentence will become part of the summary in the current agent. Thus, the c-th agent of the initial population is created as shown in Eq. (9):

$$X_c(0) = \left[x_{c,1}(0), x_{c,2}(0), \ldots, x_{c,n}(0)\right], x_{c,s}(0) = a_s \tag{9}$$

where a_s is a random value in {0,1}, c = 1,2, …, p and s = 1,2, … ,n., p is the population size and n is the number of sentences.

Evaluation (HM.Evaluate) and Optimization (HM.Optimize) of the Initial Population. After generating the initial population randomly, the fitness value of each agent is calculated using Eqs. (7) and (8). A percentage op of the population is then optimized using greedy local search, which is explained further on. Finally, the fitness is recalculated and the resulting population is ordered (**HM.Sort**) from highest to lowest based on this new fitness value. Bearing in mind that Coverage Factor needs a special normalization process based on values registered for agents in current harmony memory, minimum (min) and maximum (max) values are calculated and used to normalize values in all agents of the memory. Every time these values change, the coverage factor is recalculated and the fitness function is thus also recalculated in an incremental and efficient way.

Improvisation of a New Harmony. A new harmony is created empty, then using the original rules of the Global-best Harmony Search algorithm (memory consideration, pitch adjustment using Particle Swarm Optimization (PSO) concept, and random selection) some sentences are selected in order to be part of the new improvised version (harmony). The fitness value of this new harmony is calculated (**newHarmony. Evaluate**), and if the min or max values of coverage change, the fitness value is updated for all agents in the harmony memory. Later, the optimization (**newHarmony. Optimize**) of the new harmony occurs, only with an *op* probability (see the Greedy local optimizer section below). Finally, in order to avoid a premature convergence or loss of diversity, the algorithm ensures that only different solutions (new harmonies) will be included in the harmony memory; therefore, if newHarmony exists in the harmony memory the process is repeated.



L: maximum allowed agent length; *hms*: harmony memory size; *hmcr*: harmony memory consideration rate; *parmin*: minimum pitch adjustment rate; *parmax*: maximum pitch adjustment rate; *nofe*: current number of objective function evaluations; *mnofe*: maximum number of objective function evaluations.

```
HM.Initialize();       // Random initialization of Harmony Memory (hms agents), each meme represents
                       // the absence or presence of the sentence in the summary. Each agent must meet
                       // the length restriction (total words <= L)
HM.Evaluate();         // Calculate min-max values of coverage and Calculate fitness for all agents in HM.
HM.Optimize();         // Only a percentage of agents in HM is optimized.
HM.Sort()              // Sort based on fitness value. Best solution is HM[0]. Worst solution is HM[hms]
While nofe < mnofe do
   currentPar = parmin+(parmax-parmin)*(nofe/mnofe); // From original Global-best Harmony Search
   Do
      newHarmony.Length = 0; // New Harmony is created empty (no sentence is selected)
      While (newHarmony.Length <= L)                       // Total words <= L
         If (U(0,1) < hmcr)                                // Memory consideration rule
            i = rand(hms)                                  // Select a random position in HM
            If (U(0,1) < currentPar)                       // Pitch adjustment rule
               i=0;                                        // Position of best solution in HM
            End If
            dimension = rand(HM[i].selectedSentences)      // Select the number of an active meme
         Else                                              // Random selection rule
            dimension = rand(n);                   // Randomly select a dimension from all possibilities
         End if
         If (newHarmony[dimension] = 1) continue while;    // Ignore this dimension if it was
                                                           // previously selected
         newHarmony[dimension] = 1;                        //Active this sentence (dimension)
         newHarmony.Length += SentenceLength[dimension];
      End While
      newHarmony.Evaluate();      // Calculate fitness for new Harmony, if the min-max values of
                                  // coverage change then update fitness for all agents in HM.
      newHarmony.Optimize();      // Tries to optimize the newHarmony.
      If (nofe >= mnofe) exit while;
   While (HM.Exists(newHarmony) )
   If (newHarmony.Fitness > HM[HMS].Fitness)
      HM[hms] = newHarmony;       // Replace the worst solution in HM by newHarmony.
      HM.Sort();
   End if
End While;
Return (HM[0]); // The agent with best fitness in HM is returned;
```

Fig. 1. Scheme of the ESDS-GHS-GLO memetic algorithm

Replacement. If the new harmony has a better fitness than the worst harmony in harmony memory, the new harmony replaces the worst harmony. The harmony memory is sorted in order to define the best and the worst harmony. It should be noted that to improve the performance of the algorithm, the sorting process can be avoided and only the best and worst harmonies in memory are calculated.

Stopping Criterion. The running of the memetic algorithm terminates when the stop condition is met. The stop condition was established earlier as a maximum number of evaluations of the objective function (*mnofe* parameter). Finally, the best founded solution (harmony) is returned, i.e. the first solution in the sorted harmony memory.

3.1 Greedy Local Optimizer

Regarding local search, ESDS-GHS-GLO uses a Greedy approach [43]. Taking into account the optimization probability *(op)*, an agent is optimized a maximum number of times *(maxnumop)*, adding and removing a sentence from the summary, and controlling the number of sentences in the agent by means of Eq. (8). If the fitness value of the new agent improves on the previous agent, the replacement is made. Otherwise, the previous agent is retained. A movement is then made again in the neighborhood, repeating the previous steps (Fig. 2 summarizes the greedy search used).

```
Lss: a list of sentences sorted by a reduced version of the fitness function equal to
$f_i = \alpha \times \left(\frac{RankingPosition_i}{MaxRanking}\right) + \beta \times \left(\frac{Length_i}{MaxLength}\right) + \gamma \times SimCos\left(S_i, \overline{\overline{D}}\right)$.
op: optimization probability; maxnumop: maximum number of optimizations; OriginalAgent: original agent
(agent to optimize);
----------------------------------------------------------------------------------------------------
If (U(0,1) > op) Then Return;             // Do not optimize
For i=1 … Maxnumop do
    CurrentAgent = Copy (OriginalAgent);
    Add_sentence (CurrentAgent);          // A sentence with the highest value of the reduced fitness of
                                          // the list Lss that is not part of the current agent is activated.
    Delete_sentence (CurrentAgent);       // A sentence with the lowest value of similarity of
                                          // the list Lss is turned off in the current agent.
    Length_restriction (CurrentAgent);    // The restriction of the summary length is executed.
    Evaluate (CurrentAgent);              // Calculate fitness for current agent. If max-min values of
                                          // coverage factor change, update fitness for all agents in HM
    If (Fitness(CurrentAgent) > Fitness(OriginalAgent)) Then OriginalAgent = CurrentAgent;
End For
```

Fig. 2. Procedure of greedy local optimization

The neighborhood is generated based on a scheme of elitism in which the sentence denoted as a one (i.e. included in the candidate summary) is selected from a list sorted according to the similarity of the sentence to the document centroid; and the sentence denoted as a zero (being thus removed from the candidate summary) contains least similarity to the document centroid. This means the coverage factor is the criterion used to include or remove a sentence from the candidate summary.

4 Experiment and Evaluation

To evaluate the ESDS-GHS-GLO algorithm, Document Understanding Conference (DUC) datasets for the years 2001 and 2002 were used. These collections are a product of research by the National Institute of Standards and Technology and are available online at http://www-nlpir.nist.gov. The DUC2001 collection comprises 309 documents; and DUC2002 comprises 567 documents. In these collections, the summary generated should be less than 100 words and have several reference summaries for each document.

Pre-processing of the documents involves linguistic techniques such as segmentation of sentences or words [39], removal of stop words, removal of capital letters and punctuation marks, stemming and indexing [39]. This process is carried out before starting to run the algorithm for the automatic generation of summaries.

The segmentation process was done using an open source segmentation tool called "splitta" (available at http://code.google.com/p/splitta). Stop word removal was carried out based on the list built for the SMART information retrieval system (ftp://cs.cornell.edu/pub/smart/english.stop). The Porter algorithm was used for the stemming process. Finally, Lucene (http://lucene.apache.org) was used to facilitate the entire indexing and searching in information retrieval tasks.

Evaluation of the quality of the summaries generated was performed using metrics provided by the assessment tool ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [44] version 1.5.5 (available on internet), which has been widely handled (official metric) by DUC in evaluating automatic summaries. Because the proposed algorithm is not deterministic, the algorithm was run thirty (30) times over each document to obtain the average of each ROUGE measure.

Comparison of the proposed algorithm was made against MA-SingleDocSum [35] and DE [31] (metaheuristic approach), FEOM (fuzzy evolutionary approach) [21], UnifiedRank (graph-based approach) [17], NetSum (machine learning approach based on neural nets) [9], CRF (machine learning approach based on Conditional Random Fields) [10], QCS (machine learning approach based on Hidden Markov Model) [7], SVM (algebraic approach) [20], and Manifold Ranking (probabilistic approach using greedy algorithm) [17].

4.1 Parameter Tuning

Parameter tuning was carried out based on the Meta Evolutionary Algorithm (Meta-EA) [45], using a version of harmony search [46]. The configuration of parameters for the ESDS-GHS-GLO algorithm is as follows: Harmony memory size *hms* = 10, harmony memory consideration rate *hmcr* = 0.85, minimum pitch adjustment rate *parmin* = 0.01, maximum pitch adjustment rate *parmax* = 0.99, optimization probability *op* = 0.25, maximum number of optimizations *maxnumop* = 5 (maximum number of times an agent is optimized), maximum length of summary to evolve *mlse* = 110 (during the evolutionary process), maximum number of objective function evaluations *mnofe = 1600,* $\alpha = 0.50$, *$\beta = 0.30$, and $\gamma = 0.20$.* The algorithm was implemented on a PC Intel Core I7 3.0 GHz CPU with 12 GB of RAM in Windows 8.1.

As regards the objective function, the process of tuning the weights of the ESDS-GHS-GLO objective function was divided into two stages. In the first, a subset of all documents (DUC2001 and DUC2002) was selected as a training set. Using a Meta-EA approach based on HS the best weights for all factors were defined. In the second stage, the best weights obtained were used over all documents in order to obtain the results shown in the next section.

4.2 Results

Table 1 presents the results obtained in ROUGE-1 and ROUGE-2 measures, for ESDS-GHS-GLO and other state-of-the-art methods on the DUC2001 and DUC2002 data sets. The best solution is represented in bold type. The number in the right part of each ROUGE value in the table shows the ranking of each method. As shown in this table, MA-SingleDocSum improves upon the other methods in all ROUGE-2 measures for DUC2001 and DUC2002, and ESDS-GHS-GLO obtains second place. DE obtains best ROUGE-1 results on DUC2001 and UnifiedRank obtains best ROUGE-1 results on DUC2002.

Table 1. ROUGE values for each method on DUC2001 and DUC2002.

	DUC2001				DUC2002			
Method	ROUGE-1		ROUGE-2		ROUGE-1		ROUGE-2	
MA-SingleDocSum	0.44862	7	**0.20142**	**1**	0.48280	2	**0.22840**	**1**
ESDS-GHS-GLO	0.45402	5	0.19565	2	0.47896	3	0.22138	2
DE	**0.47856**	**1**	0.18528	4	0.46694	4	0.12368	6
FEOM	0.47728	2	0.18549	3	0.46575	5	0.12490	5
UnifiedRank	0.45377	6	0.17646	7	**0.48487**	**1**	0.21462	3
NetSum	0.46427	3	0.17697	6	0.44963	6	0.11167	7
QSC	0.44852	8	0.18523	5	0.44865	7	0.18766	4
CRF	0.45512	4	0.17327	8	0.44006	8	0.10924	8
SVM	0.44628	9	0.17018	9	0.43235	9	0.10867	9
Manifold Ranking	0.43359	10	0.16635	10	0.42325	10	0.10677	10

Because the results do not identify which method gets the best results on both data sets, a unified ranking of all methods is presented, taking into account the position each method occupies for each measure. Table 2 shows the unified ranking. The resultant rank in this table (last column) was computed according to Eq. (10)

$$Rank(method) = \sum_{r=1}^{10} \frac{(11 - r + 1) \times R_r}{10} \tag{10}$$

where R_r denotes the number of times the method appears in the r-th rank. The denominator 10 corresponds to the total number of compared methods. High values of Rank are desired.

Table 2. The resultant rank of the methods.

	$R_r =$										Rank
Methods	1	2	3	4	5	6	7	8	9	10	
MA-SingleDocSum	2	1	0	0	0	0	1	0	0	0	3.3
ESDS-GHS-GLO	0	2	1	0	1	0	0	0	0	0	3.2
DE	1	0	0	2	0	1	0	0	0	0	2.9
FEOM	0	1	1	0	2	0	0	0	0	0	2.9
UnifiedRank	1	0	1	0	0	1	1	0	0	0	2.7
NetSum	0	0	1	0	0	2	1	0	0	0	2.2
QSC	0	0	0	1	1	0	1	1	0	0	2.0
CRF	0	0	0	1	0	0	0	3	0	0	1.6
SVM	0	0	0	0	0	0	0	0	4	0	0.8
Manifold Ranking	0	0	0	0	0	0	0	0	0	4	0

Considering the results of Table 2 the following can be observed:

- The MA-SingleDocSum algorithm takes first place in the ranking (the highest value of the column Rank in the Table 2), focusing optimization on sentences position, sentences length, similarity of the sentence with the document title, cohesion and coverage of the summary. The fitness function uses five factors, and those factors are not normalized, so the weight of each factor is not in fact so meaningful.
- The ESDS-GHS-GLO method takes second place in the ranking, but MA-SingleDocSum used more execution time and uses a more complex fitness function. ESDS-GHS-GLO outperforms other methods based on metaheuristic approach (DE proposal), fuzzy evolutionary approach (FEOM), graph-based approach (UnifiedRank), machine learning approach (NetSum, QCS, and CRF), algebraic approach (SVM), and probabilistic approach using greedy algorithm (Manifold Ranking).
- The metaheuristic approach outperforms all remaining methods (machine learning, algebraic reduction, and probabilistic methods). Machine learning approach (using neural nets, conditional random fields, and hidden markov models) outperforms the algebraic and probabilistic methods. Finally, the algebraic reduction approach outperforms the probabilistic approach.

The experimental results indicate that optimization that combines global search based on population (Global-best Harmony Search) with a heuristic local search for some of the agents (greedy search) - as is the case with the ESDS-GHS-GLO memetic algorithm - is a promising area of research for the problem of generating extractive summaries for a single document. This approach is similar to previous research where a genetic algorithm was combined with guided local search, but it now features an easier and more meaningful fitness function.

5 Conclusions and Future Work

This paper proposes a new memetic algorithm for automatically generating extractives summaries from a single document - ESDS-GHS-GLO, based on Global-best Harmony Search and greedy search. For this problem, the agent is represented using many “zeros” and very few “ones” (sentences selected for the summary) but can also be implemented as a list featuring only the selected sentences. Using the Global-best Harmony Search algorithm, the design process of the algorithm is easier, because the designer does not have to worry about the selection, crossover, mutation and replacement tasks common in genetic algorithms.

The ESDS-GHS-GLO method proposed was evaluated by means of ROUGE-1 and ROUGE-2 measures on DUC2001 and DUC2002 datasets. Metaheuristic methods (including the proposed ESDS-GHS-GLO) surpass all methods in the state of the art over all measures. The best solutions are achieved by MA-SingleDocSum, ESDS-GHS-GLO, and DE. Therefore, regarding results obtained in the task of automatically generating summaries using memetic algorithms, the use of these in this type of problem is promising, but it is necessary to continue to conduct research in order to achieve better results than those obtained in this paper.

Considering possible future work, it is necessary to carry out experiments on other data sets, and to include other characteristics in the objective function that allow the selection of sentences relevant to the content of the documents and obtain a summary that is closer to the reference summaries built by humans; likewise to evaluate the use of other similarity measures such as soft cosine measure [47]; furthermore local search algorithms should also be explored, taking into account the characteristics specific to the automatic generation of summaries.

Acknowledgments. The work in this paper was supported by the University of Cauca and the National University of Colombia. We are especially grateful to Colin McLachlan for suggestions relating to English text.

References

1. Nenkova, A., McKeown, K.: A survey of text summarization techniques. In: Aggarwal, C. C., Zhai, C. (eds.) Mining Text Data, pp. 43–76. Springer, New York (2012)
2. Lloret, E., Palomar, M.: Text summarisation in progress: a literature review. Artif. Intell. Rev. **37**(1), 1–41 (2012)
3. Miranda, S., Gelbukh, A., Sidorov, G.: Generación de resúmenes por medio de síntesis de grafos conceptuales. Revista Signos. Estudios de Lingüística **47**(86) (2014)
4. Edmundson, H.P.: New methods in automatic extracting. J. ACM **16**(2), 264–285 (1969)
5. Aone, C., et al., Trainable, scalable summarization using robust NLP and Machine Learning. In: Mani, I., Maybury, M.T. (eds.) Advances in Automatic Text Summarization, pp. 71–80 (1999)
6. Kupiec, J., Pedersen, J., Chen. F.: A trainable document summarizer. In: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, Seattle, Washington, USA (1995)

7. Dunlavy, D.M., et al.: QCS: a system for querying, clustering and summarizing documents. Inf. Process. Manage. **43**(6), 1588–1605 (2007)
8. Conroy, J., O'leary, D.: Text summarization via hidden Markov models. In: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, New Orleans, Louisiana, USA (2001)
9. Svore, K., Vanderwende, L., Burges, C.: Enhancing single-document summarization by combining RankNet and third-party sources. In: Proceedings of the EMNLP-CoNLL (2007)
10. Shen, D., et al.: Document summarization using conditional random fields. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence. Morgan Kaufmann Publishers Inc., Hyderabad, India (2007)
11. Wong, K.-F., Wu, M., Li, W.: Extractive summarization using supervised and semi-supervised learning. In: Proceedings of the 22nd International Conference on Computational Linguistics. Association for Computational Linguistics, Manchester, UK (2008)
12. Marcu, D.: Improving summarization through rhetorical parsing tuning. In: Proceedings of the Sixth Workshop on Very Large Corpora, Montreal, Canada (1998)
13. Ono, K., Sumita, K., Miike, S.: Abstract generation based on rhetorical structure extraction. In: Proceedings of the 15th Conference on Computational Linguistics. Association for Computational Linguistics, Kyoto, Japan (1994)
14. Barzilay, R., Elhadad, M.: Using lexical chains for text summarization. In: Proceedings of the ACL/EACL 1997 Workshop on Intelligent Scalable Text Summarization, Madrid, Spain (1997)
15. Louis, A., Joshi, A., Nenkova, A.: Discourse indicators for content selection in summarization. In: Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue, pp. 147–156. Association for Computational Linguistics, Tokyo, Japan (2010)
16. Mihalcea, R., Tarau, P.: Text-rank: bringing order into texts. In: Proceeding of the Conference on Empirical Methods in Natural Language Processing, Barcelona, Spain (2004)
17. Wan, X.: Towards a unified approach to simultaneous single-document and multi-document summarizations. In: Proceeding of the 23rd International Conference on Computational Linguistics (Coling 2010), Beijing (2010)
18. Gong, Y.: Generic text summarization using relevance measure and latent semantic analysis. In: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (2001)
19. Steinberger, J., Jezek, K.: Using latent semantic analysis in text summarization and summary evaluation. In: Proceedings of the 7th International Conference ISIM (2004)
20. Yeh, J.-Y., et al.: Text summarization using a trainable summarizer and latent semantic analysis. Inf. Process. Manage. **41**(1), 75–95 (2005)
21. Steinberger, J., Ježek, K.: Sentence compression for the LSA-based summarizer, pp. 141–148 (2006)
22. Lee, J.-H., et al.: Automatic generic document summarization based on non-negative matrix factorization. Inf. Process. Manage. **45**(1), 20–34 (2009)
23. Dehkordi, P.-K., Kumarci, F., Khosravi, H.: Text summarization based on genetic programming. In: Proceedings of the International Journal of Computing and ICT Research (2009)
24. Qazvinian, V., Sharif, L., Halavati, R.: Summarising text with a genetic algorithm-based sentence extraction. Int. J. Knowl. Manage. Stud. (IJKMS) **4**(4), 426–444 (2008)
25. García-Hernández, R.A., Ledeneva, Y.: Single extractive text summarization based on a genetic algorithm. In: Carrasco-Ochoa, J.A., Martínez-Trinidad, J.F., Rodríguez, J.S., Baja, G.S. (eds.) MCPR 2012. LNCS, vol. 7914, pp. 374–383. Springer, Heidelberg (2013)

26. Litvak, M., Last, M., Friedman, M.: A new approach to improving multilingual summarization using a genetic algorithm. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, Uppsala, Sweden (2010)
27. Fattah, M.A., Ren, F.: GA, MR, FFNN, PNN and GMM based models for automatic text summarization. Comput. Speech Lang. **23**(1), 126–144 (2009)
28. Meena, Y.K., Gopalani, D.: Evolutionary algorithms for extractive automatic text summarization. Procedia Comput. Sci. **48**, 244–249 (2015)
29. Binwahlan, M.S., Salim, N., Suanmali, L.: Swarm diversity based text summarization. In: Leung, C.S., Lee, M., Chan, J.H. (eds.) ICONIP 2009, Part II. LNCS, vol. 5864, pp. 216–225. Springer, Heidelberg (2009)
30. Shareghi, E., Hassanabadi, L.S.: Text summarization with harmony search algorithm-based sentence extraction. In: Proceedings of the 5th International Conference on Soft Computing as Transdisciplinary Science and Technology. Cergy-Pontoise, France (2008)
31. Aliguliyev, R.M.: A new sentence similarity measure and sentence based extractive technique for automatic text summarization. Expert Syst. Appl. **36**(4), 7764–7772 (2009)
32. Abuobieda, A., Salim, N., Kumar, Y.J., Osman, A.H.: An improved evolutionary algorithm for extractive text summarization. In: Selamat, A., Nguyen, N.T., Haron, H. (eds.) ACIIDS 2013, Part II. LNCS, vol. 7803, pp. 78–89. Springer, Heidelberg (2013)
33. Binwahlan, M.S., Salim, N., Suanmali, L.: Fuzzy swarm diversity hybrid model for text summarization. Inf. Process. Manage. **46**, 571–588 (2010)
34. Song, W., et al.: Fuzzy evolutionary optimization modeling and its applications to unsupervised categorization and extractive summarization. Expert Syst. Appl. **38**(8), 9112–9121 (2011)
35. Mendoza, M., et al.: Extractive single-document summarization based on genetic operators and guided local search. Expert Syst. Appl. **41**(9), 4158–4169 (2014)
36. Garcia-Martinez, C., Rodriguez, F.J., Lozano, M.: Analysing the significance of no free lunch theorems on the set of real-world binary problems. In: 2011 11th International Conference on Intelligent Systems Design and Applications (ISDA) (2011)
37. Omran, M.G.H., Mahdavi, M.: Global-best harmony search. Appl. Math. Comput. **198**(2), 643–656 (2008)
38. Geem, Z.W.: Music-Inspired Harmony Search Algorithm: Theory and Applications. Studies in Computational Intelligence, vol. 191, 206. Springer Publishing Company, Incorporated, Rockville, Maryland (2009)
39. Manning, C., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
40. Hachey, B., Murray, G., Reitter, D.: The Embra System at DUC 2005: query-oriented multi-document summarization with a very large latent semantic space. In: Proceedings of the Document Understanding Conference (DUC), Vancouver, Canada (2005)
41. Alguliev, R.M., et al.: MCMR: Maximum coverage and minimum redundant text summarization model. Expert Syst. Appl. **38**, 14514–14522 (2011)
42. Lin, C.-Y., Hovy, E.: Identifying topics by position. In: Proceedings of the Fifth Conference on Applied Natural Language Processing, San Francisco, CA, USA (1997)
43. Ochoa, G., Verel, S., Tomassini, M.: First-improvement vs. best-improvement local optima networks of NK landscapes. In: Schaefer, R., Cotta, C., Kołodziej, J., Rudolph, G. (eds.) PPSN XI. LNCS, vol. 6238, pp. 104–113. Springer, Heidelberg (2010)
44. Lin, C.-Y.: Rouge: a package for automatic evaluation of summaries. In: Proceedings of the ACL-04 Workshop on Text Summarization Branches Out, Barcelona, Spain (2004)

45. Eiben, A.E., Smit, S.K.: Evolutionary algorithm parameters and methods to tune them. In: Monfroy, E., Hamadi, Y., Saubion, F. (eds.) Autonomous Search, pp. 15–36. Springer, Berlin (2012)
46. Cobos, C., Estupiñán, D., Pérez, J.: GHS + LEM: global-best Harmony Search using learnable evolution models. Appl. Math. Comput. **218**(6), 2558–2578 (2011)
47. Sidorov, G., et al.: Soft similarity and soft cosine measure: similarity of features in vector space model. Computación y Sistemas **18**(3) (2014)