Middle-East Journal of Scientific Research 13 (8): 1070-1086, 2013
ISSN 1990-9233

DOI: 10.5829/idosi.mejsr.2013.13.8.526

Using Gene Expression Programming in Automatic Text Summarization

Pouya Khosravian Dehkordi and Farshad Kyoumarsi

Islamic Azad University, Shahrekord Branch, Iran

Abstract: This work proposes an approach to address the problem of improving content selection in automatic text summarization by using some methods. This approach is a trainable summarizer, which takes into account several features, including Mean-TF-ISF, Sentence Length, Sentence Position, Similarity to Title, Similarity to Keywords, Sentence-to-Sentence Cohesion, Sentence-to-Centroid Cohesion, Referring position in a given level of the tree, Indicator of main concepts, Occurrence of proper nouns, Occurrence of anaphors and Occurrence of non-essential information for each sentence to generate summaries. First, we investigate the effect of each sentence feature on the summarization task. Then we use all features in combination to train gene expression programming (GEP), vector approach and fuzzy approach in order to construct a text summarizer for each model. Furthermore, we use trained models to test summarization performance. The proposed approach performance is measured at several compression rates on a data corpus composed of 100 English Reading Texts and DUC 2005. The results of the proposed approach are evaluating by ROUGE and they are promising, especially the GEP approach.

Key words: Automatic text summarization · Gene expression programming · Vector approach · Fuzzy approach

INTRODUCTION

Automatic text summarization has been an active research area for many years. Evaluation of summarization is a quite hard problem. Often, a lot of manual labour is required, for instance by having humans read generated summaries and grading the quality of the summaries with regards to different aspects such as information content and text clarity. Manual labour is time consuming and expensive. Summarization is also subjective. The conception of what constitutes a good summary varies a lot between individuals and of course also depending on the purpose of the summary.

Automatic text summarization has been studied for decades [1] and is still a very active area [2-10]. Only a few have tried using machine learning to accomplish this difficult task [2, 11, 12]. Most research falls into combining statistical methods with linguistic analysis. We regard the summarization as a problem of empowering a machine to learn from human-summarized text documents.

With the huge amount of information available electronically, there is an increasing demand for automatic text summarization systems. Text summarization is the process of automatically creating a compressed version of a given text that provides useful information for the user [13-16]. Text summarization addresses both the problem of selecting the most important portions of text and the problem of generating coherent summaries. There are two types of summarization: extractive and abstractive. Extractive summarization methods simplify the problem of summarization into the problem of selecting a representative subset of the sentences in the original documents. Abstractive summarization may compose novel sentences, unseen in the original sources. However, abstractive approaches require deep natural language processing such as semantic representation, inference and natural language generation, which have yet to reach a mature stage nowadays [17].

Recently many experiments have been conducted for the text summarization task. Some were about evaluation of summarization using relevance prediction [18], ROUGEeval package [19], SUMMAC, NTCIR and Document Understanding Conferences (DUC) [20] and voted regression model [21]. Others were about single- and multiple-sentence compression using ‘‘parse and trim” approach and a statistical noisy-channel approach [22] and conditional random fields [23]. Other research includes multi-document summarization [24, 25] and summarization for specific domains [26, 27, 28].

Corresponding Author: Pouya Khosravian Dehkordi, Islamic Azad University, Shahrekord Branch, Iran.

We employ an evolutionary algorithm, Gene Expression Programming (GEP) [29], as the learning mechanism in our Adaptive Text Summarization (ATS) system to learn sentence ranking functions. Even though our system generates extractive summaries, the sentence ranking function in use differentiates ours from that of [1, 30, 31] who specified it to be a linear function of sentence features. We used GEP to generate a sentence ranking function from the training data and applied it to the test data, which also differs from [11] who used decision tree, [2, 4] who used Bayes's rule and [12] who implemented decision tree.

In this work, sentences of each document are modeled as vectors of features extracted from the text. The summarization task can be seen as a two-class classification problem, where a sentence is labeled as ''correct" if it belongs to the extractive reference summary, or as ''incorrect" otherwise. We may give the ''correct" class a value '1' and the ''incorrect" class a value '0'. In testing mode, each sentence is given a value between '0' and '1' (values between 0 and 1 are continuous). Therefore, we can extract the appropriate number of sentences according to the compression rate. The trainable summarizer is expected to ''learn" the patterns which lead to the summaries, by identifying relevant feature values which are most correlated with the classes ''correct" or ''incorrect". When a new document is given to the system, the ''learned" patterns are used to classify each sentence of that document into either a ''correct" or ''incorrect" sentence by giving it a certain score value between '0' and '1'. A set of highest score sentences are chronologically specified as a document summary based on the compression rate.

The rest of the paper is organized as follows: Section 2 presents the different text feature parameters, Section 3 is about the proposed automatic summarization model, Section 4 shows the experimental results and finally Section 5 presents conclusions and future work.

Background: We concentrate our presentation in two main points: (1) the set of employed features; and (2) the framework defined for the trainable summarizer, including the employed classifiers.

Text Features: A large variety of features can be found in the text-summarization literature. In our proposal we employ the following set of features:

(F1) Mean-TF-ISF. Since the seminal work of [32], text processing tasks frequently use features based on IR measures. In the context of IR, some very important measures are term frequency (TF) and term frequency - inverse document frequency (TF-IDF). In text summarization we can employ the same idea: in this case we have a single document d and we have to select a set of relevant sentences to be included in the extractive summary out of all sentences in d. Hence, the notion of a collection of documents in IR can be replaced by the notion of a single document in text summarization. Analogously the notion of document - an element of a collection of documents - in IR, corresponds to the notion of sentence - an element of a document - in summarization. This new measure will be called term frequency - inverse sentence frequency and denoted (TF-ISF). The final used feature is calculated as the mean value of the TF-ISF measure for all the words of each sentence.

(F2) Sentence Length: This feature is employed to penalize sentences that are too short, since these sentences are not expected to belong to the summary [33]. We use the normalized length of the sentence, which is the ratio of the number of words occurring in the sentence over the number of words occurring in the longest sentence of the document.

(F3) Sentence Position: This feature can involve several items, such as the position of a sentence in the document as a whole, its the position in a section, in a paragraph, etc. and has presented good results in several research projects.

We use here the percentile of the sentence position in the document, as proposed by [34]; the final value is normalized to take on values between 0 and 1.

(F4) Similarity to Title: According to the vectorial model, this feature is obtained by using the title of the document as a "query" against all the sentences of the document; then the similarity of the document's title and each sentence is computed by the cosine similarity measure.

(F5) Similarity to Keywords: This feature is obtained analogously to the previous one, considering the similarity between the set of keywords of the document and each sentence which compose the document, according to the cosine similarity. For the next two features we employ the concept of text cohesion. Its basic principle is that sentences with higher degree of cohesion are more relevant and should be selected to be included in the summary. This feature must be introduced by expert person of that language.

(F6) Sentence-to-Sentence Cohesion: This feature is obtained as follows: for each sentence *s* we first compute the similarity between *s* and each other sentence *s* of the document; then we add up those similarity values, obtaining the raw value of this feature for *s*; the process is repeated for all sentences. The normalized value (in the range [0, 1]) of this feature for a sentence *s* is obtained by computing the ratio of the raw feature value for *s* over the largest raw feature value among all sentences in the document. Values closer to 1.0 indicate sentences with larger cohesion.

(F7) Sentence-to-Centroid Cohesion: This feature is obtained for a sentence *s* as follows: first, we compute the vector representing the centroid of the document, which is the arithmetic average over the corresponding coordinate values of all the sentences of the document; then we compute the similarity between the centroid and each sentence, obtaining the raw value of this feature for each sentence. The normalized value in the range [0, 1] for s is obtained by computing the ratio of the raw feature value over the largest raw feature value among all sentences in the document. Sentences with feature values closer to 1.0 have a larger degree of cohesion with respect to the centroid of the document and so are supposed to better represent the basic ideas of the document.

For the next features an approximate argumentative structure of the text is employed. It is a consensus that the generation and analysis of the complete rhetorical structure of a text would be impossible at the current state of the art in text processing. In spite of this, some methods based on a surface structure of the text have been used to obtain good-quality summaries. To obtain this approximate structure we first apply to the text an agglomerative clustering algorithm. The basic idea of this procedure is that similar sentences must be grouped together, in a bottom-up fashion, based on their lexical similarity. As result a hierarchical tree is produced, whose root represents the entire document. This tree is binary, since at each step two clusters are grouped. Five features are extracted from this tree, as follows:

(F8) Referring Position in a Given Level of the Tree (Positions 1, 2, 3 and 4): We first identify the path form the root of the tree to the node containing s, for the first four depth levels. For each depth level, a feature is assigned, according to the direction to be taken in order to follow the path from the root to s; since the argumentative tree is binary, the possible values for each position are: left, right and none, the latter indicates that *s* is in a tree node having a depth lower than four.

(F9) Indicator of Main Concepts: This is a binary feature, indicating whether or not a sentence captures the main concepts of the document. These main concepts are obtained by assuming that most of relevant words are nouns. Hence, for each sentence, we identify its nouns using a part-of-speech software. For each noun we then compute the number of sentences in which it occurs. The fifteen nouns with largest occurrence are selected as being the main concepts of the text [30]. Finally, for each sentence the value of this feature is considered "true" if the sentence contains at least one of those nouns and "false" otherwise.

(F10) Occurrence of Proper Nouns: The motivation for this feature is that the occurrence of proper names, referring to people and places, are clues that a sentence is relevant for the summary. This is considered here as a binary feature, indicating whether a sentence *s* contains (value "true") at least one proper name or not (value "false"). Proper names were detected by a part-of-speech tagger [35].

(F11) Occurrence of Anaphors: We consider that anaphors indicate the presence of non-essential information in a text: if a sentence contains an anaphor, its information content is covered by the related sentence. The detection of anaphors was performed in a way similar to the one proposed by [36]: we determine whether or not certain words, which characterize an anaphor, occur in the first six words of a sentence. This is also a binary feature, taking on the value "true" if the sentence contains at least one anaphor and "false" otherwise.

(F12) Occurrence of Non-Essential Information: We consider that some words are indicators of non-essential information. These words are speech markers such as "because", "furthermore" and "additionally" and typically occur in the beginning of a sentence. This is also a binary feature, taking on the value "true" if the sentence contains at least one of these discourse markers and "false" otherwise.

Gene Expression Programming Fundamentals

Algorithm: The GEP algorithm is described in detail in [29, 37]. The main ideas are summarized here. The first steps and the most difficult ones, of the application of GEP (and of any EA) are the problem definition, the encoding of the candidate solution and the definition of the fitness function. The encoding and the fitness functions are specific to each problem. Adequate choices

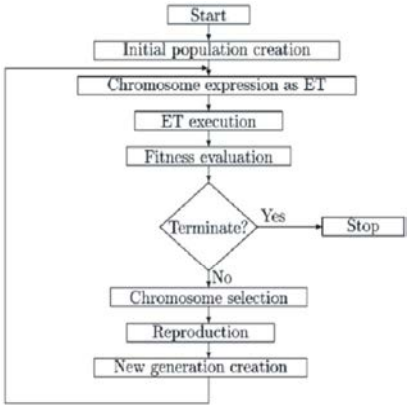


Fig. 1: Basic GEP algorithm

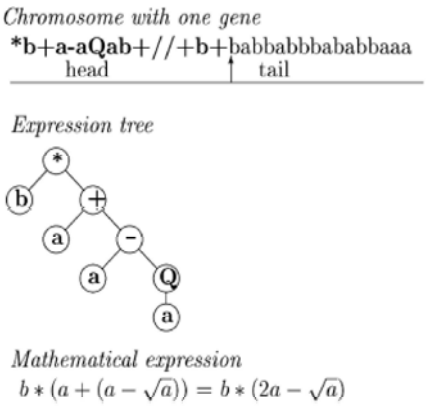


Fig. 2: Unigenic chromosome, the decoded ET and its corresponding mathematical expression

are crucial for the success of the algorithm. In making these choices knowledge about the problem and about the expected solution should be used.

The next step is the application of the GEP algorithm itself. A basic representation of the algorithm is presented in Fig. 1. Running the algorithm starts with the random creation of an initial population of chromosomes and continues with the translation of each chromosome into an Expression Tree and hence, into a mathematical expression, which is executed. The fitness function is then evaluated for each chromosome determining its fitness. Using this fitness, the termination criterion is evaluated indicating if a solution of the desired quality was found or a certain number of iterations was run. If the termination criterion is not met, some of the chromosomes are selected and reproduced, resulting in offspring. The new chromosomes will replace the old ones producing a new generation. The process continues until the termination criterion is met. The fittest chromosome is then decoded, producing the optimal solution of the problem as it was developed by the algorithm.

Chromosome Encoding: In representing the candidate solution, GEP works with two entities: the chromosome and the expression tree (ET). The chromosome is a list of functions and terminals (variables and constants) organized in one or more genes of equal length. The functions and variables are input information while the constants are created by the algorithm in a range chosen by the user. Each gene is divided into a head composed of terminals and functions and a tail composed only of terminals. The length of the head (h) is an input parameter of the algorithm while the length of the tail (t) is given by: (Eq.1)

$$t = h(n - 1) + 1$$

where n is the largest arity of the functions used in the gene's head. This head-tail partition of the gene ensures that every function of the gene has the required number of arguments available, making the chromosome correspond to a syntactically correct expression. Each gene of a chromosome is translated (decoded) into an ET with the following rules:

- The first element of the gene is placed on the first line of the ET and constitutes its root,
- On each next line of the ET a number of elements equal to the number of arguments of the functions located on the previous line is placed,
- The process is repeated until a line containing only terminals is formed.

The reverse process, the encoding of the ET into a gene, implies reading the ET from left to right and from top to bottom. An example of a chromosome with the head length equal to 15 made of five functions, *Q,*, /,*+ and -, (*Q* being the square root function) and two terminals, a and b, is shown in Fig. 2, together with its decoded ET and the corresponding mathematical expression.

It can be noticed that the ET ends before the end of the gene. This shows that the GEP genes can have non-expressed regions, just like biological genes which can have regions non-expressed in proteins. This additional information encoded in the gene is used during genetic variation whenever needed in order to create syntactically correct structures (ETs). In the case of multigenic chromosomes, the ETs corresponding to each gene are connected with a linking function defined by the user. The mathematical expression associated with these combined ETs is the candidate solution to the problem.

Reproduction: The reproduction of the chromosomes is done through two mechanisms: elitism and reproduction with modification. Elitism is the process through which the fittest chromosome is replicated unchanged into the next generation, preserving the best material from one generation to another.

Reproduction with modification is the process through which the chromosomes are selected and modified with genetic operators producing offspring. It is important to emphasis that the genetic operators are applied on the chromosomes and not on the expression trees, as in GP [38]. This fact, together with the head-tail organization of the genes, makes GEP always produce syntactically correct structures during the evolution process.

The chromosome selection for reproduction is done with the roulette-wheel [37] method. This method allows the fitter individuals to have a higher probability of producing offspring. In contrast with GA [39] and GP [38] which use mainly recombination and mutation operators, GEP has three classes of genetic operators for reproduction with modification: mutation, transposition and recombination [29].

The mutation operator randomly changes an element of a chromosome into another element, preserving the rule that the tails contain only terminals. In the head of the gene a function can be changed into another function or terminal and vice versa. In the tail a terminal can only be changed into another terminal.

The transposition operator randomly moves a part of the chromosome to another location in the same chromosome. In GEP there are three kinds of transposable elements:

- Short fragments with a function or a terminal in the first position which transpose into the head of genes, except at the root. A sequence with the same number of elements is deleted from the end of the head in order to maintain the structural organization of the gene.
- Short fragments with a function in the first position that transpose to the root of the gene. A sequence with the same number of elements is deleted from the end of the gene head.
- An entire gene that transposes to the beginning of the chromosome.

The recombination or cross-over operator exchanges parts of a pair of randomly chosen chromosomes. In GEP there are three kinds of recombination:

- One-point recombination in which the parent chromosomes are paired and split up at the same point. The material after the recombination point is exchanged between the two chromosomes, forming two new daughter chromosomes.
- Two-point recombination in which the parent chromosomes are paired and two points are randomly chosen where the chromosomes are split. The material between the recombination points is exchanged between the two chromosomes, forming two new daughter chromosomes.
- Gene recombination in which entire genes are exchanged between two parent chromosomes, forming two new daughter chromosomes containing genes from both parents.

Each genetic operator is applied on the chromosomes of a population with a certain probability called the operator rate. The values of the operator rates are chosen and optimized by the user. Mutation is the most powerful operator, creating diversity in the population. Its rate, defined as the probability of each element of the entire population to be mutated, is recommended to have small values (0.01-0.1) [37]. Additional diversity is created with the transposition operator while the recombination operator has a homogeneous effect. The transposition and recombination rates are defined as the probability of each chromosome of a population to be subject to that operator. Moderate rates are recommended for these two operators (0.1-0.4) [37].

Document Understanding Conferences (DUC): In DUC 2001-2004 a growing number of research groups participated in the evaluation of generic and focused summaries of English newspaper and newswire data. Various target sizes were used (10-400 words) and both single document summaries and summaries of multiple documents were evaluated (around 10 documents per set). Summaries were manually judged for both content and readability. To evaluate content, each peer (human or automatic) summary was compared against a single model summary using SEE (http://www.isi.edu/cyl/SEE/) to estimate the percentage of information in the model that was covered in the peer. Additionally, automatic evaluation of content coverage using ROUGE [40] was explored in 2004.

The focus of DUC 2005 was on developing new evaluation methods that take into account variation in content in human-authored summaries. Therefore, DUC 2005 had a single user-oriented, question-focused

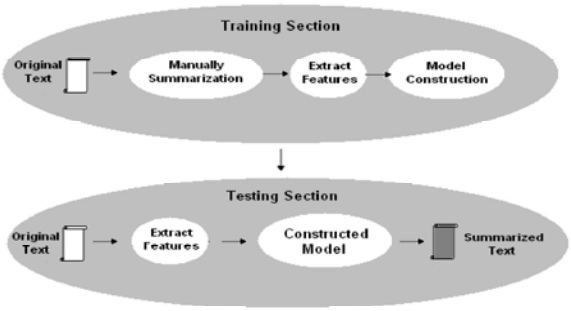


Fig. 3: The proposed automatic summarization model

summarization task that allowed the community to put some time and effort into helping with the new evaluation framework. The summarization task was to synthesize from a set of 25-50 documents a well-organized, fluent answer to a complex question. The relatively generous allowance of 250 words for each answer reveals how difficult it is for current summarization systems to produce fluent multi-document summaries.

DUC 2005 marked a major change in direction from previous years. The road mapping committee had strongly recommended that new tasks be undertaken that were strongly tied to a clear user application. Consequently, a report-writing task based on a "natural disaster" scenario was proposed at the DUC 2004 workshop, but this was met with little enthusiasm in the community. At the same time, the program committee wanted to work on new evaluation methodologies and metrics that would take into account variation of content in human-authored summaries.

Therefore, DUC 2005 had a single simpler (but still user-oriented) system task that allowed the community to put some time and effort into helping with a new evaluation framework. The system task modeled real-world complex question answering [41]. Systems were to synthesize from a set of 25-50 documents a brief, well-organized, fluent answer to a need for information that could not be met by just stating a name, date, quantity, etc. Summaries were evaluated for both content and readability.

The Proposed Automatic Summarization Model: Fig. 3 shows the proposed automatic summarization model. We have two modes of operations:

- Training mode where features are extracted from 100 manually summarized English documents and used to train Gene expression programming, Fuzzy and Vector models.
- Testing mode, in which features are calculated for sentences from 10 English documents. (These documents are different from those that were used for training). The sentences are ranked according to the sets of feature weights calculated during the training stage. Summaries consist of the highest-ranking sentences.

Experimental Results

The English Data: One hundred English reading texts in the domain of religion were collected from the Internet archive. One hundred English reading texts were manually summarized using compression rate of 30%. These manually summarized reading texts were used to train the previously mentioned models. The other 10 English reading texts were used for testing. The average number of sentences per English reading texts is 65.8, respectively. Moreover, to investigate the proposed approaches performance on newswire data, we have exploited DUC 2005 [42] for single document test.

We use an intrinsic evaluation to judge the quality of a summary based on the coverage between it and the manual summary. We measure the system performance in terms of precision from the following formula: (Eq. 2).

$$P = \frac{|S \cap T|}{|S|}$$

Recall would be calculated as: (Eq.3)

$$R = \frac{|S \cap T|}{|S|}$$

where P is the precision, R is the recall, T is the manual summary and S is the machine-generated summary.

Summarization Based on Vectorial Approach: A frequently employed text model is the vectorial model [35]. After the preprocessing step each text element - a sentence in the case of text summarization -is considered as a N-dimensional vector. So it is possible to use some metric in this space to measure similarity between text elements. The most employed metric is the cosine measure, defined as cos θ = (<x.y>) / (|x|. |y|) for vectors x and y, where (<,>) indicates the scalar product and |x| indicates the module of x. Therefore maximum similarity corresponds to cos θ = 1, whereas cos θ =0 indicates total discrepancy between the text elements. This algorithm is shown in Fig. 4.

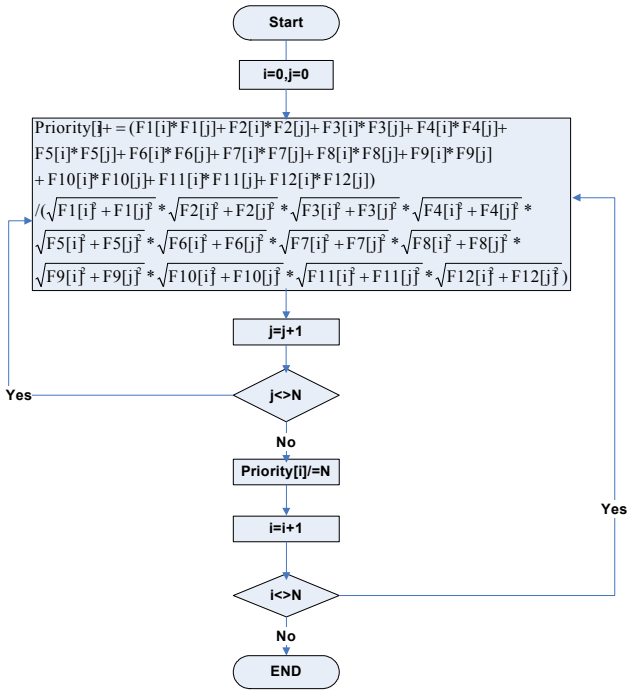


Fig. 4: The Vectorial algorithm to determine each sentence priority and N is number of sentence

Table 1: Vector Model Performance Result

Best Fitness	301.21
Max Fitness	1000
Accuracy	37.62%

Some of the attributes which are used in vector method such as, indicator of main concept, the occurrence of proper nouns and the Occurrence of non-essential information are binary parameters (0 and 1) that do not seem to work well in all situations. For example one of these attributes is "indicator of main concept". If a sentence contains at least one of these specified words, this attribute (indicator of main concept) has the value of one for that sentence; otherwise, it has the value of zero. As it is obvious, the sentence which contains one specified word has less value than the sentence which contains two specified words. This matter is ignored in ordinary methods and is considered as a shortcoming for those methods. The results of vector model performance are given in Table 1 and priority chart is shown in Fig. 5.

Text Summarization Based on Fuzzy Approach: To solve the problem of vector approach, we can use the fuzzy logic. To achieve this end, instead of using 0 and 1 for the specified attributes of the text, we can use fuzzy values which are between 0 and 1 [43].

In order to implement text summarization based on fuzzy logic, we used MATLAB (Fig. 6) since it is possible to simulate fuzzy logic in this software [43]. To do so; first, we consider each characteristic of a text such as sentence length, location in paragraph, similarity to key word and etc, which was mentioned in the text feature part, as the input of Mandani style fuzzy system. Then, we enter all the rules needed for summarization, in the knowledge base of this system. After ward, a value from zero to one is obtained for each sentence in the output based on sentence characteristics and the available rules in the knowledge base. The obtained value in the output determines the degree of the importance of the sentence in the final summary. This operation is shown in Fig. 7.

The results of fuzzy model performance are given in Table 2 and priority chart is shown in Fig. 8.

Text Summarization Based on Gene Expression Programming Approach: We are going to exploit the GEP approach of [29, 37], for summarization and use it as a baseline approach. We use the approach for testing, a set of 100 English documents was used. All document sentences are ranked in a descending order according to their scores. A set of highest score sentences are chronologically specified as a document summary based on the compression rate.

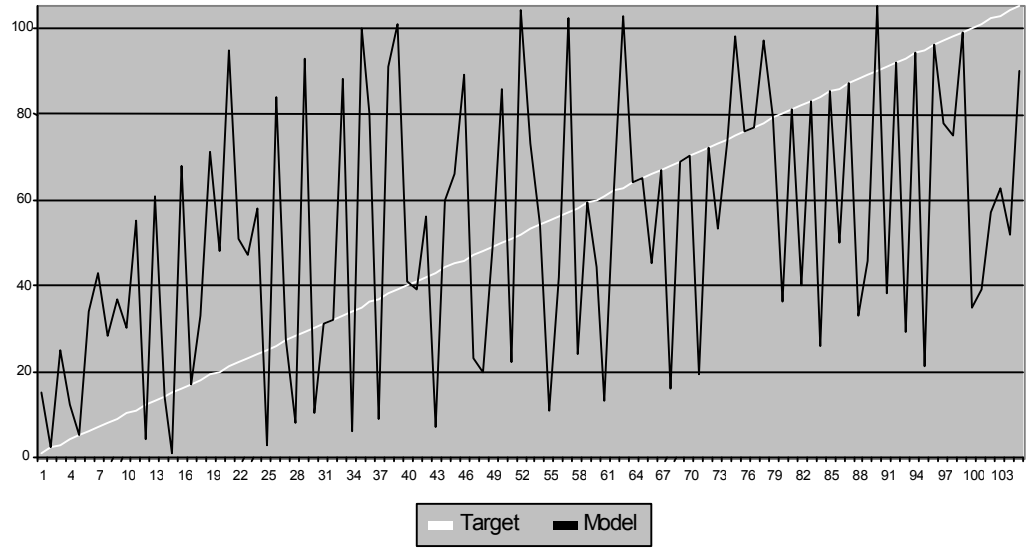


Fig. 5: This chart shows wrong sentence priority of Vector Model than Human Target in testing level

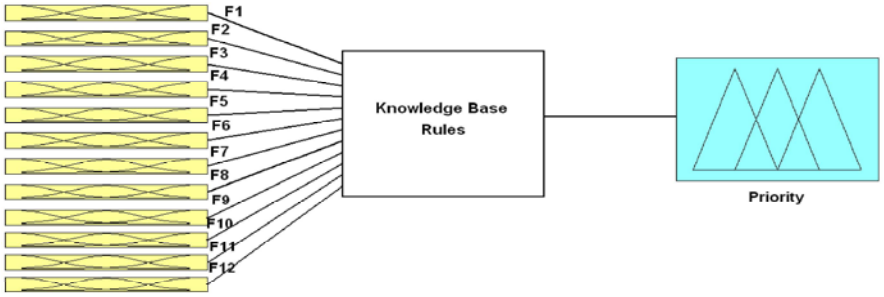


Fig. 6: MATLAB schema for text feature analyzing

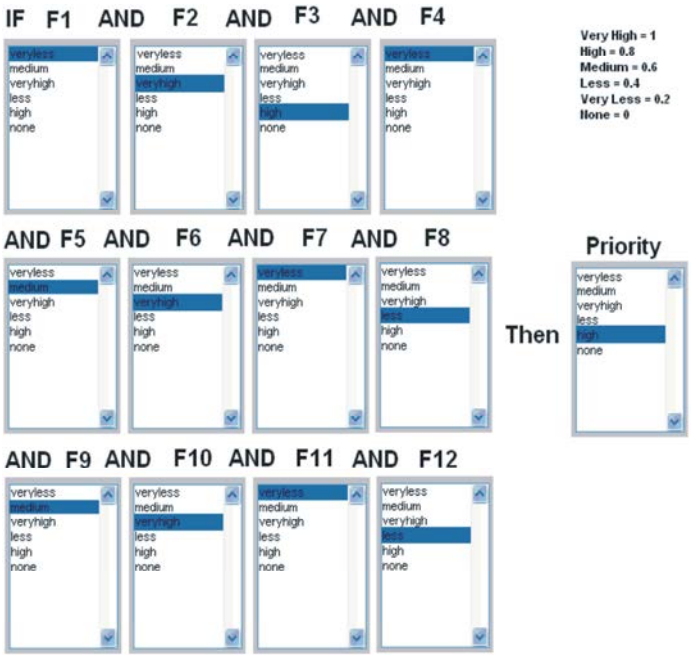


Fig. 7: The fuzzy operation to determine each sentence priority

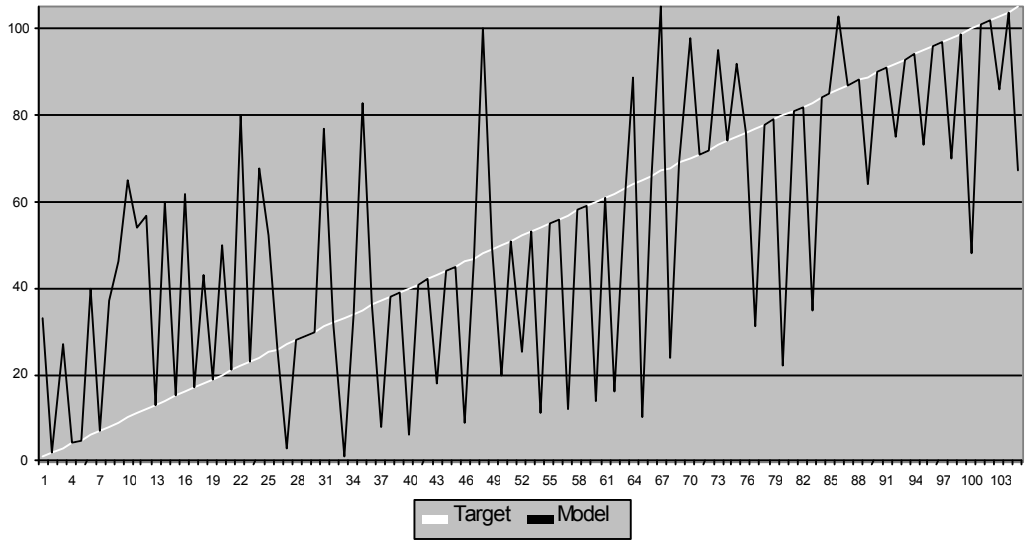


Fig. 8: This chart shows wrong sentence priority of Fuzzy Model than Human Target in testing level

Table 2: Fuzzy Model Performance Result

Best Fitness	540.73
Max Fitness	1000
Accuracy	57.29%

Table 3: GEP Data

Independent Variables:	12
Training Samples:	1016
Testing Samples:	105

Table 4: GEP Program Structure

Program Size:	87
Literals:	67
Used Variables:	F1(5), F10(1), F11(3), F12(6), F2(3), F3(2), F4(3), F5(3), F6(3), F7(3), F8(4), F9(2)

Table 5: GEP General Settings

Chromosomes:	30
Genes:	4
Head Size:	10
Tail Size:	31
Dc Size:	31
Gene Size:	72
Linking Function:	Addition

Table 6: GEP Genetic Operators

Mutation Rate:	0.044
Inversion Rate:	0.1
IS Transposition Rate:	0.1
RIS Transposition Rate:	0.1
One-Point Recombination Rate:	0.3
Two-Point Recombination Rate:	0.3
Gene Recombination Rate:	0.1
Gene Transposition Rate:	0.1

Table 7: GEP Numerical Constants

Constants per Gene:	2
Data Type:	Floating-Point
Lower Bound:	-10
Upper Bound:	10
RNC Mutation:	0.01
Dc Mutation:	0.044
Dc Inversion:	0.1
Dc IS Transposition:	0.1

Gene Expression Programming Configuration: Related parameters for the training and testing of the GEP model like Data, Program Structure, general setting, genetic operators and Numerical Constants are given in Table 3, 4, 5, 6 and 7.

Gene Expression Programming Model: We have exploited the GEP approach of Ferreira [29], for summarization as described above using GEP Model. Therefore, we have exploited the twelve features for summarization. The system calculates the feature weights using gene expression programming.

All document sentences are ranked in a descending order according to their scores. A set of highest score sentences are chronologically specified as a document summary based on the compression rate. Tables 10 and 11 show the GEP approach performance evaluation based on precision and recall using the twelve features for English documents, respectively. To do this we using GEP finding function model [38].

GEP Model Explicit Formulation: By using GEP Finding Function model and analyzing data we got GEP Model and model results are given in Table 8:

Generation	Program Size	Literals	Used Variables	Training Fitness	Training Accuracy
407879	87	67	F1(5), F10(3), F11(1), F12(6), F2(3), F3(2), F4(4), F5(3), F6(5), F7(3), F8(6), F9(2)	1000	100.00%

```
GOE3H.Sin.LT3D.LOE2A.LT3G.Cosh.NET2C.3Rt.Csc.LT3D.c1.d6.d8.d1.d6.d0.d10.d3.d0.d3.c1.d9.d6.d2.d5.d8.d8.d1.c1.d5.c1
 .c0.d5.c0.d1.c0.d10.d7.c0.d8.c0
+
GT3H.Sin.GOE3H.LOE2A.LT3G.Cosh.Sin.ET3I.GOE2F.LT4B.d0.d0.d8.d3.d6.d7.d3.d7.c0.d10.c1.c0.d0.d8.d2.d1.d2.d4.d7.d3.d1
 .c0.c1.c0.c1.c1.d4.c0.c1.c1.c0
+
NET2G.GOE4K.Sin.NET2G.d9.GOE3K.GOE3J.d2.Cosh.Logi2.d3.d2.d2.d4.d5.d9.d4.c1.c0.c0.d7.c1.c0.d10.c0.d10.d11.d2.d2.c1.d8
 .d9.c0.c1.c0.d0.d7.d7.d10.d2.c0
+
LOE3B.ET4K.Min3.ET3A.GOE3A.d11.Sin.Tan.GT2E.AND3.d9.d5.c0.c0.d10.c1.d10.d11.d4.d3.c1.d7.c0.c1.d2.d1.c0.d10.c1.d7.d2.d8
  .d5.c0.d1.c0.d10.d10.c1.d6.c1

Numerical Constants:

Gene 1          Gene 2          Gene 3          Gene 4
c0 = -2.597717  c0 = -4.726227  c0 = -4.726227  c0 = 4.307739
c1 = -6.947754  c1 = -2.16513   c1 = -0.208282  c1 = -6.947754
```

Table 8: Specify data was produced by GEP concepts for automatic text summarization. The *di* is a selected text feature from dataset and *ci* is a numerical constant

Table 9: GEP function set for producing ETs

Definition	Function	symbol
If x+y>=z then x*y else x-z	Greater Or Equal To with 3 inputs (H)	GOE3H
Sin(x)	Sine	Sin
If x+y<z then x+y else x-z	Less Than with 3 inputs (D)	LT3D
If x<=y then x else y	Less Or Equal To with 2 inputs (A)	LOE2A
If x+y<z then x*y else x+z	Less Than with 3 inputs (G)	LT3G
Cosh(x)	Hyperbolic cosine	Cosh
If x!=y then x+y else x-y	Not Equal To with 2 inputs (C)	NET2C
x^(1/3)	Cube root	3Rt
Csc(x)	Cosecant	Csc
If x+y<z then x+y else x-z	Less Than with 3 inputs (D)	LT3D
If x+y>z then x*y else x-z	Greater Than with 3 inputs (H)	GT3H
If x+y=z then x*y else x*z	Equal To with 3 inputs (I)	ET3I
If x>=y then x+y else sin(x*y)	Greater Or Equal To with 2 inputs (F)	GOE2F
If a+b<c+d then c else d	Less Than with 4 inputs (B)	LT4B
If x!=y then x+y else atan(x*y)	Not Equal To with 2 inputs (G)	NET2G
If a+b<c+d then sin(a*b) else sin(c*d)	Greater Or Equal To with 4 inputs (K)	GOE4K
If x+y>=z then x+y+z else sin(x*y*z)	Greater Or Equal To with 3 inputs (K)	GOE3K
If x+y>=z then x*y else x/z	Greater Or Equal To with 3 inputs (J)	GOE3J
Logistic(x,y)	Logistic(x,y)	Logi2
If x+y<=z then x+y else z	Less Or Equal To with 3 inputs (B)	LOE3B
If a+b=c+d then sin(a*b) else sin(c*d)	Equal To with 4 inputs (K)	ET4K
Min(x,y,z)	Minimum of 3 inputs	Min3
If x=0 then y else z	Equal To with 3 inputs (A)	ET3A
If x>=0 then y else z	Greater Or Equal To with 3 inputs (A)	GOE3A
Tan(x)	Tangent	Tan
If x>y then x+y else x*y	Greater Than with 2 inputs (E)	GT2E
If x<=0 AND y<=0 then 1 else 0	AND3	AND3

By using Table 8 and function set (Table 9), we can produce Expression Trees (ETs) like Fig. 9:

Evaluation GEP Model: We used 100 English text documents for training and 10 English text documents for testing GEP model and the results are given in Table 10 and 11. The model priority chart is shown in Fig. 10.

The Effect of Each Feature on Summarization Performance: In this section, we investigate the effect of each feature parameter on summarization by using GEP model with individual score using feature weight. For instance, to investigate the first feature (sentence position) on summarization performance, we use the following equation: (Eq.4).

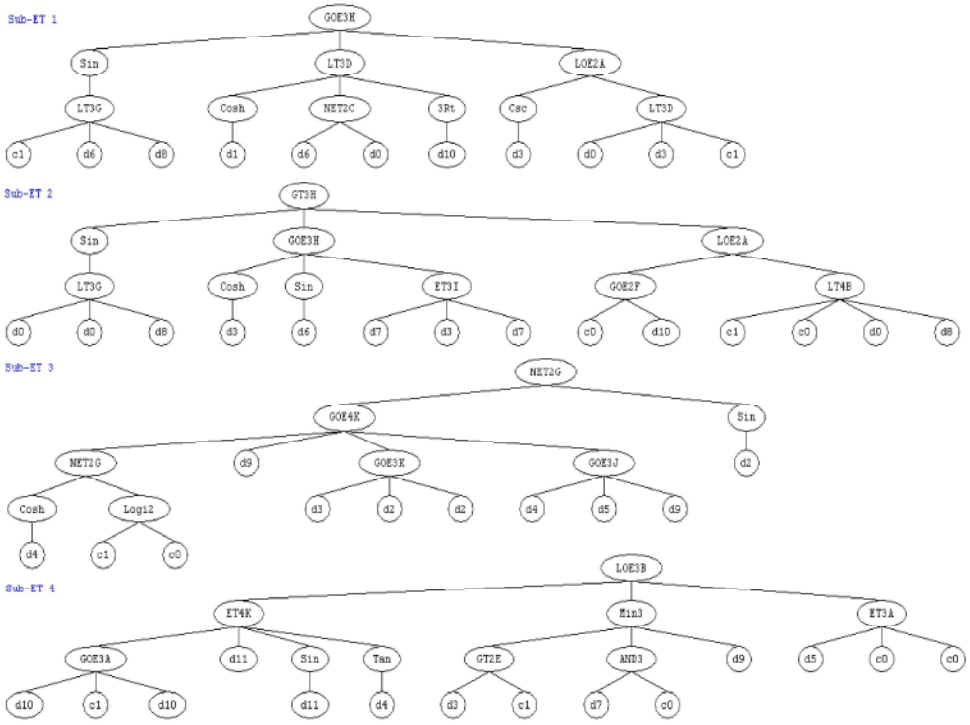


Fig. 9: ETs for automatic text summarization based on GEP

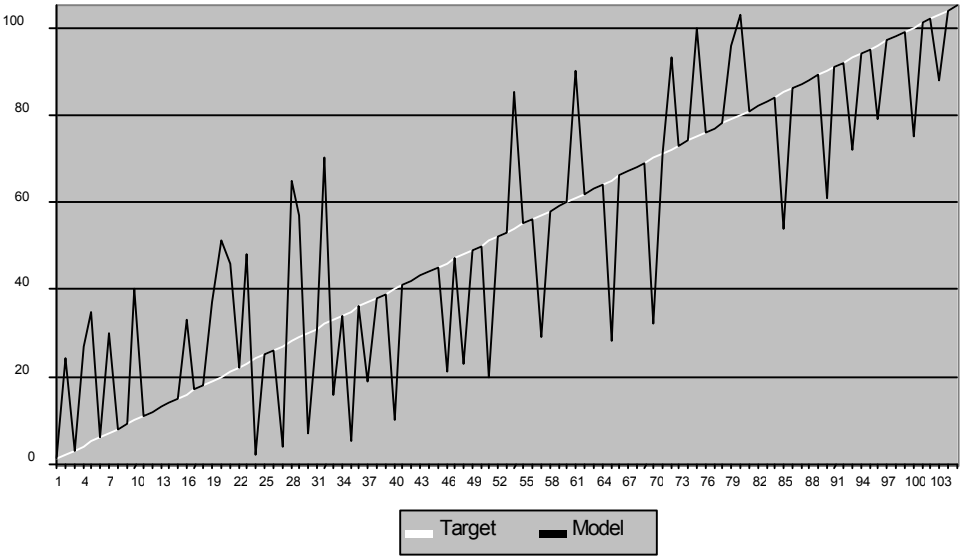


Fig. 10: This chart shows wrong sentence priority of GEP Model than Human Target in testing level.

Table 10: Statistics - Training

Best Fitness:	1000
Max. Fitness:	1000
Accuracy:	100.00%

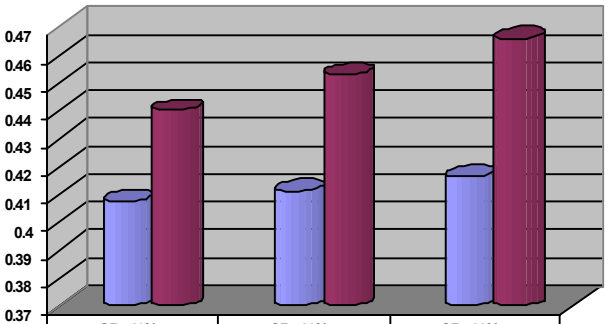
Table 11: Statistics - Testing

Best Fitness:	684.8485
Max. Fitness:	1000
Accuracy:	68.01%

$$Score(F1) = Score_{f_1}(GEPModel)$$

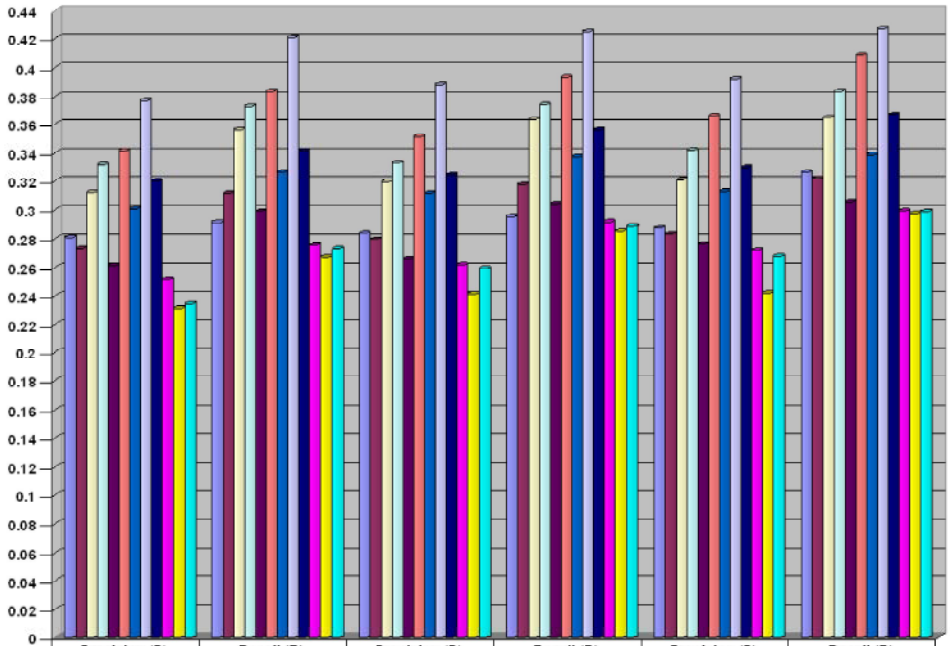
Fig. 11 and 12 show the summarization precision associated with each feature for different compression rates for English documents respectively.

Rouge Evaluation: Traditionally evaluation of summarization involves human judgments of different



	CR 10%	CR 20%	CR 30%
Precision	0.4076	0.4113	0.4164
Recall	0.4403	0.453	0.466

Fig. 11: The GEP approach performance evaluation based on precision and recall for different compression rate (CR)



	Precision (P)	Recall (R)	Precision (P)	Recall (R)	Precision (P)	Recall (R)
F1	0.2809	0.2911	0.2841	0.2953	0.2879	0.326
F2	0.2734	0.3113	0.2798	0.3177	0.2834	0.3213
F3	0.312	0.3564	0.3192	0.3636	0.3206	0.365
F4	0.3315	0.3732	0.3328	0.3745	0.3417	0.3834
F5	0.2609	0.2988	0.2658	0.3037	0.2764	0.3055
F6	0.3413	0.3834	0.3513	0.3934	0.3663	0.4084
F7	0.3005	0.3262	0.3114	0.3371	0.3127	0.3384
F8	0.3775	0.4209	0.3892	0.425	0.3923	0.4271
F9	0.3201	0.341	0.3245	0.3562	0.3298	0.3674
F10	0.2513	0.2757	0.2617	0.2916	0.2722	0.2991
F11	0.2313	0.2674	0.2413	0.2857	0.2423	0.2973
F12	0.2347	0.2731	0.2589	0.2889	0.2676	0.2986

Fig. 12: The summarization precision and recall associated with each feature for different compression rates (CR)

quality metrics, for example, coherence, conciseness, grammaticality, readability and content [6]. However, even simple manual evaluation of summaries on a large scale over a few linguistic quality questions and content coverage as in the Document Understanding Conference (DUC) [23] would require over 3,000 hours of human efforts. This is very expensive and difficult to conduct in a frequent basis. Therefore, how to evaluate summaries automatically has drawn a lot of attention in the summarization research community in recent years. For example, [44] proposed three content-based evaluation methods that measure similarity between summaries. These methods are: *cosine similarity*, *unit overlap* (i.e. unigram or bigram) and *longest common*

subsequence. However, they did not show how the results of these automatic evaluation methods correlate to human judgments. Following the successful application of automatic evaluation methods, such as BLEU [45], in machine translation evaluation, [45] showed that methods similar to BLEU, i.e. n-gram co-occurrence statistics, could be applied to evaluate summaries.

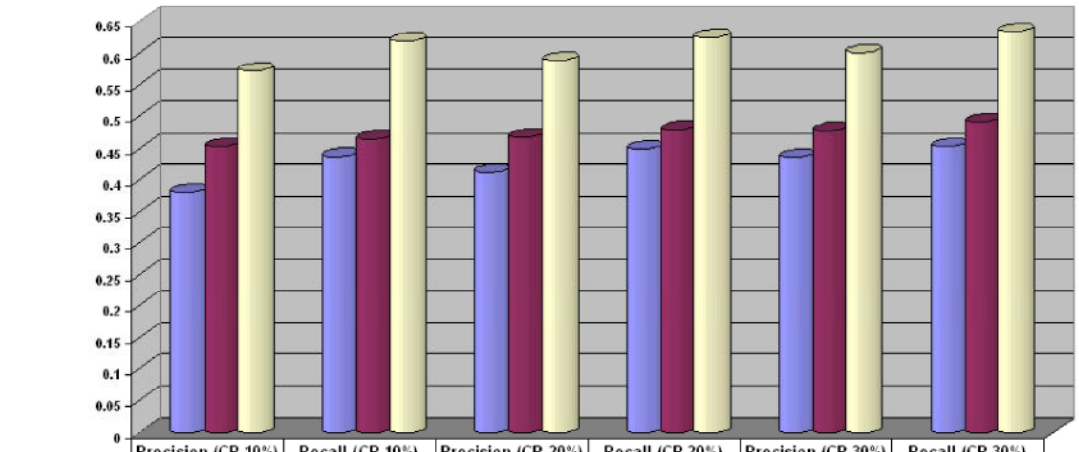
ROUGE-N: N-Gram Co-Occurrence Statistics: Formally, ROUGE-N is an n-gram recall between a candidate summary and a set of reference summaries. ROUGE-N is computed as follows: (Eq.5).

$$\text{ROUGE-N}=\frac{\sum_{S\in\{Referemce\ Summaries\}}\ \sum_{gram_n\in S} Count_{match}(gram_n)}{\sum_{S\in\{Referemce\ Summaries\}}\ \sum_{gram_n\in S} Count_{match}(gram_n)}$$

where *n* stands for the length of the n-gram, *gramn* and *Countmatch*(*gramn*) is the maximum number of n-grams co-occurring in a candidate summary and a set of reference summaries. It is clear that ROUGE-N is a recall-related measure because the denominator of the equation is the total sum of the number of n-grams occurring at the reference summary side. A closely related measure, BLEU, used in automatic evaluation of machine translation, is a precision-based measure. BLEU measures how well a candidate translation matches a set of reference translations by counting the percentage of n-grams in the candidate translation overlapping with the references. Please see [45] for details about BLEU.

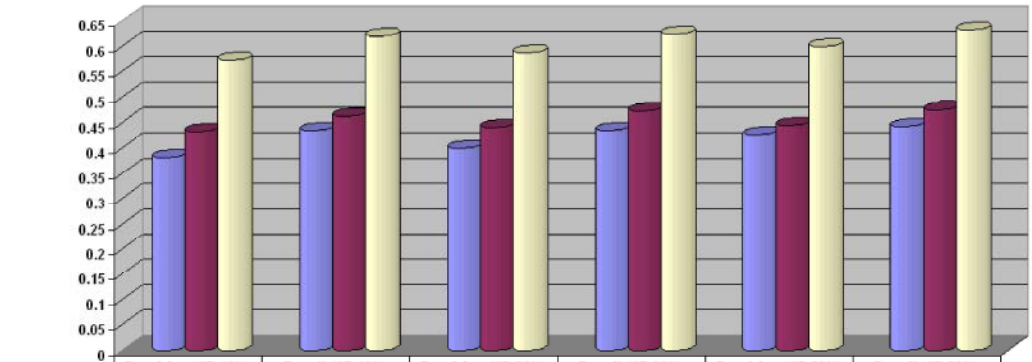
Note that the number of n-grams in the denominator of the ROUGE-N formula increases as we add more references. This is intuitive and reasonable because there might exist multiple good summaries. Every time we add a reference into the pool, we expand the space of alternative summaries. By controlling what types of references we add to the reference pool, we can design evaluations that focus on different aspects of summarization. Also note that the numerator sums over all reference summaries.

The results of all models trained on English data and tested on DUC 2005 data: In this experiment, we train all previously mentioned models on the general text features (using the same 100 English reading texts) and test these models on the DUC 2005 data [42] to investigate the proposed system performance on a newswire data. Fig. 13 shows the results of all models for the 100 English reading texts. We have created new extractive reference summaries of the DUC 2005 testing data by measuring the similarity (vocabulary overlap) between each sentence and the associated reference single document summary. Then we rank each document sentences based on this similarity value. A set of sentences is specified as a reference summary for each document based on the compression ratio. Fig. 14 shows the results of all models for the DUC 2005 testing data based on precision and recall. Fig. 15 shows the results of all models for the DUC 2005 testing data based on the average Rouge-1 score.



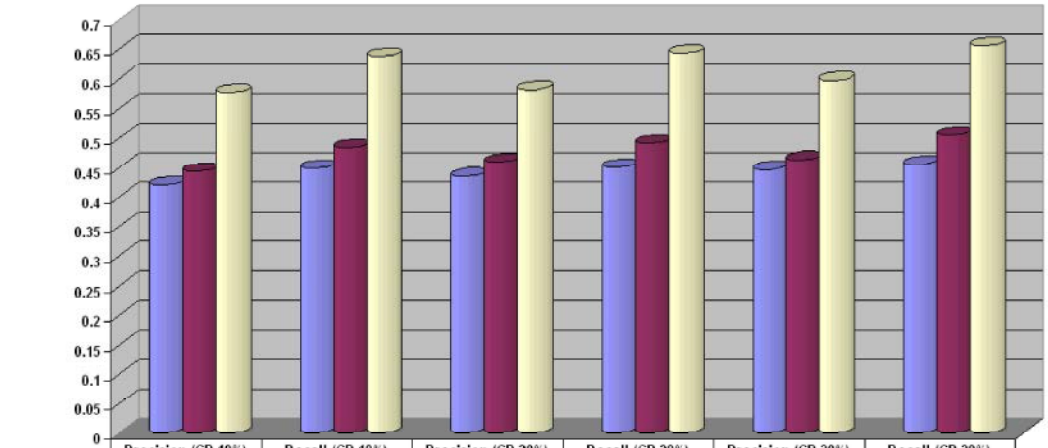
	Precision (CR 10%)	Recall (CR 10%)	Precision (CR 20%)	Recall (CR 20%)	Precision (CR 30%)	Recall (CR 30%)
Vector Model	0.3799	0.4362	0.4111	0.4478	0.4348	0.4514
Fuzzy Model	0.4521	0.4645	0.4669	0.4782	0.4774	0.4895
GEP Model	0.5708	0.6179	0.5873	0.6234	0.5998	0.6327

Fig. 13: All models performance evaluation based on precision and recall for different compression rate (English testing data)



	Precision (CR 10%)	Recall (CR 10%)	Precision (CR 20%)	Recall (CR 20%)	Precision (CR 30%)	Recall (CR 30%)
Vector Model	0.3802	0.4336	0.3998	0.4344	0.426	0.4427
Fuzzy Model	0.4312	0.4624	0.4397	0.4718	0.4449	0.4749
GEP Model	0.5712	0.6183	0.5858	0.6217	0.5979	0.6312

Fig. 14: All models performance evaluation based on precision and recall for different compression rate (DUC 2005 testing data)



	Precision (CR 10%)	Recall (CR 10%)	Precision (CR 20%)	Recall (CR 20%)	Precision (CR 30%)	Recall (CR 30%)
Vector Model	0.4203	0.4492	0.4356	0.4507	0.4463	0.4551
Fuzzy Model	0.4435	0.4842	0.4593	0.4918	0.4639	0.5056
GEP Model	0.5776	0.6377	0.5824	0.6427	0.5968	0.6553

Fig. 15: All models performance evaluation based on the average Rouge-1 score for different compression rate (DUC 2005 testing data)

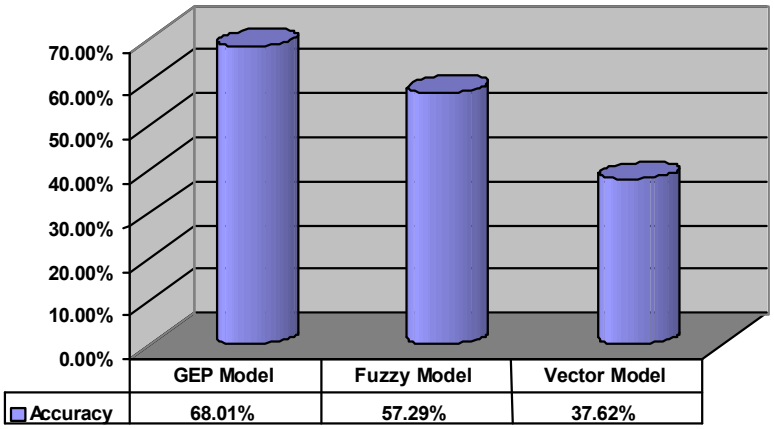


Fig. 16: The comparison of total system performance between all models

DISCUSSION

It is clear from Fig. 12 that the most important text feature for summarization is F8 (Referring position in a given level of the tree) since it gives the best results. It is reasonable, since the sentence that has a maximum number of branches should convey the most important part in the article. F6 (Sentence-to-Sentence Cohesion) also gives good results since it conveys the vocabulary overlap between this sentence and other sentences in the document. Usually, the document title conveys the main topic of this document. Therefore, F4 (Similarity to Title) which is the vocabulary overlap between this sentence and the document title gives good results. The lowest results are associated with F11 (Occurrence of anaphors) since most of reading texts do not contain many anaphors data. Therefore, the system ranks a sentence that does not contain anaphors data according to its position.

Moreover, it is clear from Table 15 that this approach can be extended to the genre of newswire text. Fig. 16 shows the total system performance of all models for English reading texts, respectively. It is clear from the figures that GEP approach gives the best results since GEP has a good capability to model arbitrary densities. The Fuzzy model has better precision than the Vector model.

CONCLUSIONS

In this paper, we have investigated the use of gene expression programming (GEP), vector approach and fuzzy approach for automatic text summarization task. We have applied our new approaches on a sample of 100 English reading texts. Our approach results outperform the baseline approach results. Our approaches have been used the feature extraction criteria which gives researchers opportunity to use many varieties of these features based on the text type.

In the future work, we will extend this approach to multi-document summarization by addressing some anti-redundancy methods which are needed, since the degree of redundancy is significantly higher in a group of topically related reading texts than in an individual article as each article tends to describe the main point as well as necessary shared background.

REFERENCES

1. Edmundson, H.P., 1969. New methods in automatic extracting. Journal of the ACM (JACM), 16(2): 264-285.
2. Aone, C., J. Gorlinsky, B. Larsen and M.E. Okurowski, 1999. A Trainable Summarizer with Knowledge Acquried from Robust NLP Techniques, *Advances in Automatic Text Summarization*, pages 71-80. The MIT Press, Cambridge, Massachusetts.
3. Brandow, R., K. Mitze and L.F. Rau, 1995. Automatic condensation of electronic publications by sentence selection. Information Processing and Management, 31(5): 675-685.
4. Kupiec, J., J. Pedersen and F. Chen, 1995. A trainable document summarizer. In: Proceedings of the 18thAnnual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'95), Seattle, WA, USA, pp: 68-73.
5. Lin, C., 1999. Training a Selection Function for Extraction. In the 8th International Conference on Information and Knowledge Management (CIKM 99), Kansa City, Missouri.
6. Mani, I., 2001. Automatic Summarization, John Benjamins Publishing Company, Amsterdam/ Philadelphia.
7. McKeown, K.R., R. Barzilay, D. Evans, V. Hatzivassiloglou, M.Y. Kan, B. Schiffman and S. Teufel, 2001. Columbia Multi-Document Summarization: Approach and Evaluation, in Proceedings of the Document Understanding Conference (DUC01). Edmonton, Canada.
8. Radev, D.R., S. Teufel, H. Saggion, W. Lam, J. Blitzer, H. Qi, A. Celebi, D. Liu and E. Drabek, 2003. Evaluation challenges in large-scale document summarization, in Proc. 41st Annual Meeting of the Association for Computational Linguistics, pp: 375-382. Sapporo, Japan.
9. Russell, S.J. and P. Norvig, 1995. Artificial Intelligence: A Modern Approach. Prentice-Hall International Inc., Englewood Cliffs, NJ. G. Salton, A. Singhal, M. Mitra and C. Buckley, Automatic text structuring and summarization. Information Processing and Management, 33(2): 193-207.
10. Salton, G., J. Allan, C. Buckley and A. Singhal, 1994. Automatic Analysis, Theme Generation and Summarization of Machine-Readable Texts. Science, 264(3): 1421-1426.
11. Lin, C.Y. and E.H. Hovy, 2003. Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics. In Proceedings of 2003 Language Technology Conference (HLT-NAACL 2003), Edmonton, Canada.
12. Neto, J.L., A.A. Freitas and C.A.A. Kaestner, 2002. Automatic Text Summarization using a Machine Learning Approach. In Proc. 16th Brazilian Symp. on Artificial Intelligence (SBIA-2002). Lecture Notes in Artificial Intelligence 2507: 205-215. Springer-Verlag.

13. Diaz, A. and P. Gerva´s, 2007. User-model based personalized summarization. Information Processing and Management, 43(6): 1715-1734.
14. Dorr, B. and T. Gaasterland, 2007. Exploiting aspectual features and connecting words for summarization-inspired temporal-relation extraction. Information Processing and Management, 43(6): 1681-1704.
15. Steinberger, J., M. Poesio, M. Kabadjov, K. Jez˘ek, 2007. Two uses of anaphora resolution in summarization. Information Processing and Management, 43(6): 1663-1680.
16. Ye, J.S., H.T. Chua, W.M. Kan, I.L. Qiu, Yeh Ye *et al*., 2007. Document concept lattice for text understanding and summarization. Information Processing and Management, 43(6): 1643-1662.
17. Yeh, S.J., T.H. Ke, M.W. Yang, L.I. Meng, Ye Yeh *et al*., 2005. Text summarization using a trainable summarizer and latent semantic analysis. Information Processing and Management, 41(1): 75-95.
18. Hobson, S., B. Dorr, C. Monz and R. Schwartz, 2007. Task-based evaluation of textsummarization using relevance prediction. Information Processing and Management, 43(6): 1482-1499.
19. Sjöbergh, J., 2007. Older versions of the ROUGEeval summarization evaluation system were easier to fool. Information Processing and Management, 43(6): 1500-1505.
20. Over, P., H. Dang and D. Harman, 2007. DUC in context. Information Processing and Management, 43(6): 1506-1520.
21. Hirao, T., M. Okumura, N. Yasuda and H. Isozaki, 2007. Supervised automatic evaluation for summarization with voted regression model. Information Processing and Management, 43(6): 1521-1535.
22. Zajic, D., B. Dorr, J. Lin and R. Schwartz, 2007. Multi-candidate reduction: sentence compression as a tool for document summarization tasks. Information Processing and Management, 43(6): 1549-1570.
23. Nomoto, T., 2007. Discriminative sentence compression with conditional random fields. Information Processing and Management, 43(6): 1571-1587.
24. Harabagiu, S., A. Hickl and F. Lacatusu, 2007. Satisfying information needs with multi-document summaries. Information Processing and Management 43(6): 1619-1642.
25. Vanderwende, L., H. Suzuki, C. Brockett and A. Nenkova, 2007. Beyond SumBasic: task-focused summarization with sentence simplification and lexical expansion. Information Processing and Management, 43(6): 1606-1618.
26. Ling, X., J. Jiang, X. He, Q. Mei, C. Zhai and B. Schatz, 2007. Generating gene summaries from biomedical literature: a study of semistructured summarization. Information Processing and Management, 43(6): 1777-1791.
27. Moens, M., 2007. Summarizing court decisions. Information Processing and Management, 43(6): 1748-1764.
28. Reeve, L., H. Han and A. Brooks, 2007. The use of domain-specific concepts in biomedical text summarization. Information Processing and Management, 43(6): 1765-1776.
29. Ferreira, C., 2001. Gene Expression Programming: A New Adaptive Algorithm for Solving Problems, Complex Systems, 13(2): 87-129.
30. Goldstein, J., M. Kantrowitz, V. Mittal and J. Carbonell, 1999. Summarizing text documents: sentence selection and evaluation metrics. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99), Berkeley, CA, USA, pp: 121-128.
31. Sekine, S. and C. Nobata, 2001. Sentence Extraction with Information Extraction technique. In Proc. Of ACM SIGIR'01 Workshop on Text Summarization. New Orleans.
32. Luhn, H.P., 1958. The automatic creation of literature abstracts. IBM Journal of Research and Development 2(2): 159-165.
33. Ferrier, L., 2001. A Maximum Entropy Approach to Text Summarization, School of Artificial Intelligence, Division of Informatics, University of Edinburgh.
34. Nevill-Manning, C.G., I.H. Witten, G.W. Paynter, *et al*., 1999. KEA: Practical Automatic Keyphrase Extraction. ACM DL, pp: 254-255.
35. Salton, G. and C. Buckley, 1988. Term-weighting approaches in automatic text retrieval. Information Processing and Management 24: 513-523. Reprinted in: Sparck-Jones.
36. Strzalkowski, T., G. Stein, J. Wang and B. Wise, 1999. A Robust Practical Text summarizer. In: I. Nani and M. Maybury, (eds.), Adv. in Autom. Text Summarization. The MIT Press.

37. Ferreira, C., 2006. Gene Expression Programming: Mathematical Modeling by an Artificial Intelligence, 2nd Edition, Springer-Verlag, Germany.
38. Koza, J.R., 1992. Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge.
39. Michalewicz, Z., 1996. Genetic Algorithms + Data Structures = Evolution Programs, Springer, Berlin.
40. Lin, C.Y., 2004. Looking for a Few Good Metrics: ROUGE and its Evaluation. In *Proceedings of NTCIR Workshop 2004*, Tokyo, Japan.
41. Amigo, E., J. Gonzalo, V. Peinado, A. Penas and F. Verdejo, 2004. An empirical study of information synthesis tasks. In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, pp: 207-214, Barcelona, Spain.
42. DUC, 2005. Document understanding conferences. http://duc.nist.gov/.
43. Khosravi, H., E. Eslami, K. Kyoomarsi and P.K. Dehkordy, 2008 Optimizing Text Summarization Based on Fuzzy Logic. In: Book Series Studies in Computational Intelligence Publisher Springer Berlin / Heidelberg, ISSN 1860-949X (Print) 1860-9503 (Online), Volume Volume 131/2008, Book Computer and Information Science, Copyright 2008, ISBN 978-3-540-79186-7, DOI 10.1007/978-3-540-79187-4_11, Pages 121-130, Subject Collection Engineering, SpringerLink Date Wednesday, May 07, 2008.
44. Saggion H., D. Radev, S. Teufel and W. Lam, 2002. Meta-Evaluation of Summaries in a Cross-Lingual Environment Using Content-Based Metrics. In *Proceedings of COLING-2002*, Taipei, Taiwan.
45. Papineni, K., S. Roukos, T. Ward and W.J. Zhu, 2001. BLEU: a Method for Automatic Evaluation of Machine Translation. IBM Research Report RC22176 (W0109-022).