HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

GRADUATION THESIS

Enhanced Genetic Algorithm for Single Document Extractive Summarization

NGUYEN TRA MY
tramy96tq@gmail.com
Computer Science

Advisor: Dr. Nguyen Thi Thu Trang

Department: Software Engineering
School: Information and Communication Technology

HA NOI, 12/2019

Requirements for the thesis

1. Student information

Student name: Tra My Nguyen
Tel: 0972401408 Email: tramy96tq@gmail.com
Class: LTU13B-K59 Program: SIE
This thesis is performed at: room 914, Ta Quang Buu Library

2. Goal of the thesis

This thesis focuses on effectively solving the problems related to extractive text summary and the development of enhanced genetic algorithm for single document extractive summarization.

3. Main tasks

In this thesis, I will present the genetic algorithm and how to improve it in order to create the optimal summary. I also use the sentence features for the selective summary.

4. Declaration of student:

I – *Tra My Nguyen* - hereby warrants that the work and presentation in this thesis are performed by myself under the supervision of *Dr. Nguyen Thi Thu Trang*. All results presented in this thesis are truthful and are not copied from any other work. All references in this thesis - including images, tables, figures, and quotes - are clearly and fully documented in the bibliography. I will take full responsibility for even one copy that violates school regulations.

Hanoi, 27th December 2019
Author

Nguyen Tra My

Acknowledgement

Foremost, I would like to express my sincere gratitude to my advisor, Dr. Nguyen Thi Thu Trang for the continuous support of my studies and researches, for her patience, motivation, enthusiam, and immense knowledge. This thesis and project would not have been possible without her timely guidance despite of her busy schedule.

Besides my supervisor, I would like to take this opportunity to special thank to Dr. Bui Thi Mai Anh, who have spent her precious time for supporting, giving me advices and creating the best conditions in the process of my project.

My sincere thanks also goes to my fellow labmate in VBEE laboratory, Nguyen Hoang Ky for all the supporting, for all the time companion, support and help me not only in works but also in normal life.

Finally, I would like to thank my family who always support and inspire me througout my life. Without their support, love and caring, I could not imagine that I can complete this thesis. I also want to send my thanks to my best friend Pham Anh Duc, who always stay beside me, support me in all up and down moments.

Abstract

Automatic summarization has enjoyed wide popularity in natural language processing due to its potential for various information access applications as tools that aid users navigate and digest web content (e.g., news, social media, product reviews), question answering, etc. Single document summarization is perhaps the most basic of summarization tasks that have been identified over the years. In extractive summarization, summaries are generated by selecting the most salient sentences from the original text. The text summarization can be seen as a classification of sentences into two groups: in-summary or not-in-summary. In recent years, various approaches for extracting key sentences have been proposed. Among many methods, extracting key sentences using the Genetic Algorithm has become a promising candidate. In this thesis, I propose an enhanced genetic algorithm in order to improve the quality of extractive text summarization. More concisely, I first evaluate the role of some sentence features and their contribution to improving the fitness function. I second investigate some crossover and mutation mechanisms in order to augment the accuracy of summarization as well as the performance of our model. The experiment has been conducted for the Daily Mail dataset to assess the proposed model and unsupervised method and supervised method as TextRank and SummaRunNer (State-of-the-art). The empirical results show that the proposed GA gives better accuracy in comparison with TextRank and SummaRunNer, increasing the accuracy by 7.2% and 6.9% respectively.

The solution in this thesis presented in this thesis was published as a regular paper in the proceedings of the **ACM international Soict 2019**- B. T. M. Anh, N. T. My, and N. T. T. Trang, "**Enhanced Genetic Algorithm for Single Document Extractive Summarization**," p. 7, 2019.

Table of contents

List of figures

List of tables

List of equation

Acronyms

GA	Genetic Algorithm
NLP	Natural Language Processing
TF-IDF	Term Frequency - Inverse Document Frequency
TF-ISF	Term Frequency - Inversed Sentence Frequency
TF	Term Frequency
IDF	Inverse Document Frequency
LCS	Longest Common Subsequence
TS	Topic Similarity
SL	Sentence Length
SP	Sentence Position
NPN	Number of Proper Nouns

Chapter 1 Introduction

1.1 Motivation

Text summarization is the process of automatically generating summaries from an input source document while retaining important points. The main goal of a summary is to present the key concepts and main ideas of the document in less space. As the number of online information increases, systems that can automatically summarize documents become increasingly desirable, for example, summarization used in search engines, report generation systems, etc [18].

Text summarization techniques can be broadly grouped into extractive and abstractive approaches. Extractive methods produce summaries by identifying important sections of the text and combining extracted parts coherently. Abstractive methods, in contrast, may generate novel words or phrases which are not in the source text - as human-written summaries. While the former is strongly concerned with the most salient sentences from the input document, the latter requires advanced language generation techniques such as paraphrasing, generalizing, incorporating real-world knowledge, etc. The abstractive-based approach, therefore, is much more complex than the extractive one. Due to the difficulty of the abstractive summarization, most research on text summarizing is extractive-based [16, 11, 13]. The extractive text summarization can be seen as a classifying task in which the sentences of the source document are categorized into two groups: summary and not-summary. Currently, there are two approaches for the classification/clustering: supervised and unsupervised learning. Supervised learning methods require large training corpus (i.e., human-generated summaries) in which all the sentences in the source document should be labeled. The labeling task may spend a lot of time and human effort. Since it is hard to get such a large corpus, there is not much research following this approach. Indeed, many works on text summarization are based on unsupervised

learning [10, 23]. Instead of using a large training corpus, these approaches use some methods to discover relations among sentences inside a document. Each sentence then will be assigned with some score. The resulting summary involves the sentences with the highest score. Various techniques have been devised for this task such as graph-based models like Text Rank [16], optimization techniques like evolutionary algorithms [4, 14]. This thesis mainly focuses on formalizing the clustering of sentences in a document as an optimal searching problem using Genetic Algorithm. Genetic Algorithm (GA) imitates the process of a natural evolution in order to solve optimization and searching problems [8]. It involves the following components:

- Representation of individuals or chromosomes, i.e., in the case of text summarization, a string which represents a sentence in the document.
- Fitness function.
- Selection of population.
- Operators such as selection, crossover, and mutation

1.2 Objective and scope

In this thesis, I aimed for three main objectives. The first is to keep yourself and understand the current common text summary methods, capturing the knowledge in the field of natural language processing. Secondly, I want to experiment to compare them and make comments. Thirdly I want to improve Genetic Algorithm and application it for extractive text summarization.

The fitness function in GA plays an important role in finding out the best results. In the field of text summarization, the fitness function is often built on the basis of sentence features such as sentence location, title similarity, sentence similarity, word similarity among sentences, word similarity among paragraphs, etc [14]. In this thesis, I aim to evaluate the salience of some sentence features in order to improve the fitness function. My goal is to investigate which sentence features contribute most when dealing with the selection of the best individuals from the population. I then aim to improve the crossover and mutation mechanisms to augment the accuracy of the resulting summarization while reducing the computation time of GA.

1.3 Structure of thesis

For the rest of this thesis is organized as follows.

Chapter 2 gives a brief overview of the background knowledge for solutions and methods mentioned in the thesis. I present text summarization, the extractive, and abstractive text summary. Then I presented the genetic algorithm as well as its components. I also talk about the equations which I use for my model.

In Chapter 3, I present the basic steps of the proposed GA to automatically generate summaries of single documents. First I will talk about the general model of genetic algorithm for text summaries. I would like to present commonly used features to evaluate the quality of a summary. These are surface features, content features, event features, relevance features. Then I select some features for my model.

I use performing uniform crossover and mutation probability for extractive summarization. Furthermore, I also adopted the precision/recall measures for automatic extractive summarization evaluation.

I also present the related work in this section.

Afterward, Chapter 4 provides information about the evaluation metric, the work environment and the data which I use. I also presented the usage parameters for my model. Lastly, I compare the results with related studies including unsupervised and supervised on the same dataset.

My conclusion about the proposed model is drawn in the Chapter 5 and give some future works.

Chapter 2 Background

In order to understand the model that I propose, let's first understand the basic concepts in an automatic text summary. This knowledge will be the foundation for the next chapters. In this chapter, I present the background knowledge for the solutions and methods mentioned in the thesis. Firstly, I introduce text summarization and classification. Next, I will briefly present Genetic Algorithms. Finally, I would like to be talking about a few used equation to my model, and its improvements to be present in Chapter 3.

2.1 Text summarization

Article abstract plain text is the same as the other summary article, is an automatic summary process with input is a document, the output is a brief piece of the document describing the main content of the head document. A single document can be a website, a post on a social network, an article, a document (e.g. doc, txt). A plain text summary is a step as the basis for the processing of multi-text summaries and more complex summary issues. That's the reason why the first-ever text summary methods are the single- document summary methods. Methods aimed at resolving single-document summary issues also focus on two summary types: extractive summarization and abstractive summarization [18].

The automated text summary is the use of computer science methods to analyze the content of the text and then produce another document that is shorter than the original text size but still ensures the content of the information is the importance of the original text.

The computer science methods here largely uses the technique to convert the text in the form of human-readable on the form of machines that can be read and processed quickly as a vector, matrix, etc. Then apply the methods of processing on those digital

data, eventually producing digitized results and reverse conversion to a readable human text.

A summary is shorter than the original document, and the higher the compression (i.e. the shorter the summary), the greater the loss of information is often. Therefore, the goal of developing methods of summarizing a text is to make the length as small as possible but minimize the loss of information as much as possible.

In addition to the assessment of the amount of retained information and the amount of information lost, it is also important to evaluate a system, a method of the summary is effective or not. Hence the world some "measure" for the evaluation of summarized text has been studied, born. Typically it is a measure of Rouge [12, 24] (this measure will be learned and used later in section 4.1.2).

Based on the aspects, the purpose for which people divide the text into different categories. Based on the amount of text that you want to summarize, it is divided into plain text summary and multi-text summary. The Summary menu text is summarized from a plain text into a different text of a smaller length. The multi-text summary is summarized from multiple original texts into a text of a smaller length.

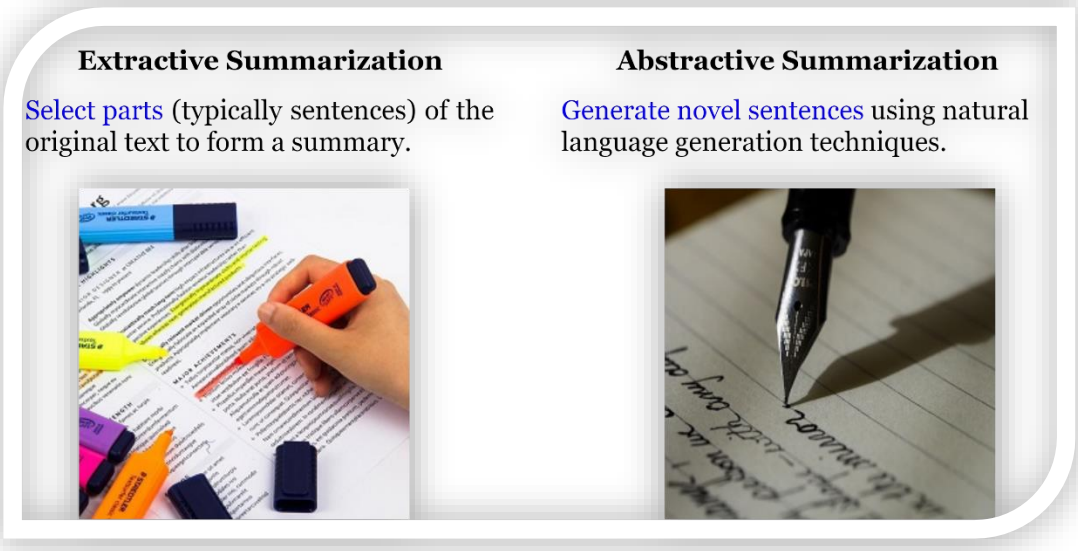

Figure 2.1 Two approaches to summarization.

Based on the specificity of the summary, it is divided into Abstractive summarization (2.1.1) and Extractive summarization (2.1.2). Extractive summarization is a summary that removes some sentences, retains some sentences, while the content, structure, grammar of each sentence remains intact. Abstractive summarization is the using natural language generation to rewrite the sentence, the contents of the sentence are retained to some extent but the grammar and the previous order of the words in that sentence may be changed.

2.1.1 Abstractive summarization

The abstractive method is a summary method based on the meaning, semantics, content of a document. The approaches can be mention as relying on information extraction, ontology, and compressed information, etc. One of the summary methods in the good results are methods based on the extraction of information, this method uses predefined templates about an event or plot, and the system will automatically fill in the available form information and generate the summary result. Despite the good results, however the methods this form usually only applies in a given domain.

In this summary method, sentences will be rewritten in an order other than the original sentence but still try to keep its meaning, content, and grammar. So this method is also much harder than the method of extracting summary.

2.1.2 Extractive summarization

The extractive summarization is the method of removing away units of sentences. The unit may be the word, maybe the sentence, maybe the passage, etc. Removed units are often the unit does not make sense of information in that document, the units that are retained are the units of important information, the core that will be retained in the summary. The units that are retained are the units with a strong connection to each other in the text, which is present in the position or the content of their coherent in the text.

The basic steps of the excerpt summary method include a waiver of redundant words (e.g. from concatenation, frequent repetition and non-significant information),

applying the algorithm to ranking units in the text. Then pick out the highest-ranked units in a given proportion. To connect the units, I will get a summary of a smaller size than the original.

In the text excerpt summary, we can divide into smaller summary methods according to algorithms: Surface level approach algorithm, medium-level approach algorithm, deep analysis algorithm.

2.2 Genetic Algorithm

Genetic algorithms [8, 20] are among the most popular evolutionary algorithms in terms of the diversity of their applications. A vast majority of well-known optimization problems have been tried by genetic algorithms. In addition, genetic algorithms are population-based, and many modern evolutionary algorithms are directly based on genetic algorithms or have some strong similarities.

The genetic algorithm, developed by John Holland and his collaborators in the 1960s and 1970s, is a model or abstraction of biological evolution based on Charles Darwin's theory of natural selection. Holland was probably the first to use the crossover and recombination, mutation, and selection in the study of adaptive and artificial systems. These genetic operators form an essential part of the genetic algorithm as a problem-solving strategy. Since then, many variants of genetic algorithms have been developed and applied to a wide range of optimization problems, from graph coloring to pattern recognition, from discrete systems (such as the traveling salesman problem) to continuous systems (e.g., the efficient design of airfoil in aerospace engineering), and from financial markets to multiobjective engineering optimization. There are many advantages of genetic algorithms over traditional optimization algorithms. Two of the most notable are the ability to deal with complex problems and parallelism. Genetic algorithms can deal with various types of optimization, whether the objective (fitness) function is stationary or nonstationary (changes with time), linear or nonlinear, continuous or discontinuous, or with random noise. Because multiple offsprings in a population act like independent agents, the population (or any subgroup) can explore the search space in

many directions simultaneously. This feature makes it ideal to parallelize the algorithms for implementation. Different parameters and even different groups of encoded strings can be manipulated at the same time. However, genetic algorithms also have some disadvantages. The formulation of a fitness function, the use of population size, the choice of important parameters such as the rate of mutation and crossover, and the selection criteria of the new population should be carried out carefully. Any inappropriate choice will make it difficult for the algorithm to converge or it will simply produce meaningless results. Despite these drawbacks, genetic algorithms remain one of the most widely used optimization algorithms in modern nonlinear optimization.

The essence of GA involves the encoding of an optimization function as arrays of bits or character strings to represent chromosomes, the manipulation operations of strings by genetic operators, and the selection according to their fitness, with the aim to find a good (even optimal) solution to the problem concerned. This is often done by the following procedure:

(i) encoding the objectives or cost functions;

(ii) defining a fitness function or selection criterion;

(iii) creating a population of individuals;

(iv) carrying out the evolution cycle of iterations by evaluating the fitness of all the individuals in the population, creating a new population by performing crossover and mutation, fitness-proportionate reproduction, etc, and replacing the old population and iterating again using the new population.

(v) decoding the results to obtain the solution to the problem.

These main steps in producing the genetic algorithms given shown by Koza (1995) in Figure 2.2.

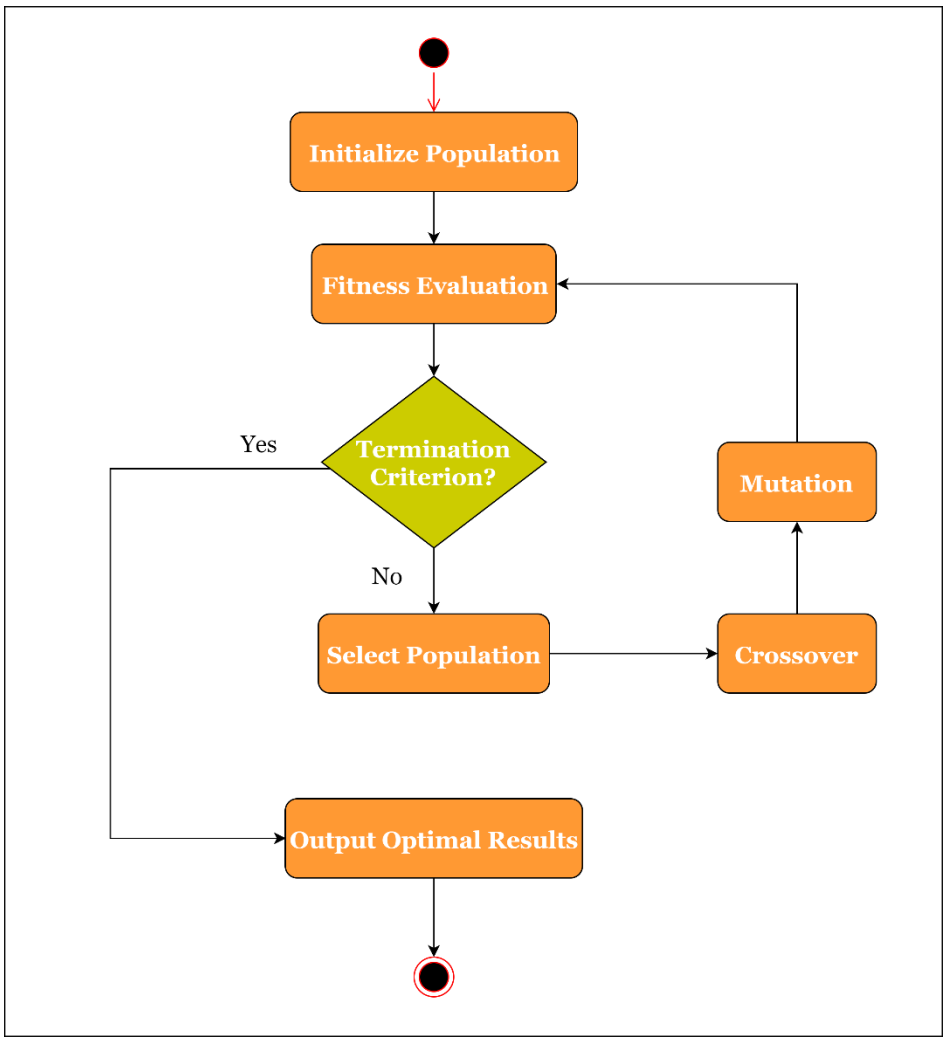

Figure 2.2 The basic structure of Genetic Algorithm [8].

2.2.1 Role of Genetic Operators

As introduced earlier, genetic algorithms have three main genetic operators: crossover, mutation, and selection. Their roles can be very different.

• Crossover: Swapping parts of the solution with another in chromosomes or solution representations. The main role is to provide mixing of the solutions and convergence in a subspace.

• Mutation: The change of parts of one solution randomly, which increases the diversity of the population and provides a mechanism for escaping from a local optimum.

• Selection of the fittest, or elitism: The use of the solutions with high fitness to pass on to the next generations, which is often carried out in terms of some form of a selection of the best solutions.

Obviously, in actual algorithms, the interactions between these genetic operators make behavior very complex. However, the role of the individual components remains the same.

Crossover is mainly an action with a subspace. This point becomes clear for a binary system where the strings consist of a and b. For example, for two strings S1 = [aabb] and S2 = [abaa], whatever the crossover actions will be, their offsprings will always be in the form [a...]. That is, the crossover can only result in solutions in a subspace where the first component is always a. Furthermore, two identical solutions will result in two identical offspring, no matter how the crossover has been applied. This means that crossover works in a subspace and the converged solutions/states will remain converged.

On the other hand, mutation usually leads to a solution outside the subspace. For the previous example, if the mutation occurs on the first a and it flips to b, then the solution S3 = [babb] does not belong to the previous subspace. In fact, mutation typically generates solutions that may be further from current solutions, thus increasing the diversity of the population. This will enable the population to escape from a trapped local optimum. One important issue is random selection among the population. For example, crossover requires two parents in the population. Do we choose them randomly or biased toward the solutions with better fitness? One way is to use a roulette wheel to do the selection; another is to use fitness-proportional selection. Obviously, there are other forms of selection in use, including linear ranking selection, tournament selections, and others.

Both crossover and mutation work without the use of the knowledge of the objective or fitness landscape. Selection of the fittest, or elitism, on the other hand, does use the fitness landscape to guide what to choose and thus affects the search behavior of an algorithm. What is selected and how solutions are selected depends on the

algorithm as well as the objective function values. This elitism ensures that the best solutions must survive in the population. However, very strong elitism may lead to premature convergence. It is worth pointing out that these genetic operators are fundamental. Other operators may take different forms, and hybrid operators can also work. However, to understand the basic behavior of genetic algorithms, I will focus on these key operators in 2.2.2, 2.2.3.

2.2.2 Initialization

Traditionally, binary strings of 0s and 1s are used to represent an individual of the population, but other encodings are also possible for example decimal. The evolution usually starts from a population of randomly generated individuals and happens in generations. The reproduction operators select chromosomes from the population to be parents and conducting the crossover process. In each generation, the fitness of every individual in the population is evaluated. The selection of a chromosome for parenthood can range from a totally random process to one that is biased by the chromosome's fitness. The fitness evaluation returns the value to the fitness of an individual in GA. This evaluation function judges the quality of chromosomes or fitness to solve a problem. The fitness evaluation function acts as an interface between the GA and the optimization problem. First, decode the chromosome, and then use the fitness function to evaluate. The fitness function returns a value indicating the fitness of chromosomes to solve the problem. The results of the fitness evaluation function determine the probability that a possible solution is selected to produce new solutions in the next generation.

2.2.3 Crossover

In the reproduction operator, two chromosomes selected from the population-based on its fitness. Then, copy the individual into the next generation of the population without change. Reproduction operator established into the population in two main ways: crossover and mutation. Crossover is an interchange of parts of two individuals. The crossover operator creates two offspring chromosomes, which contain some genetic material of its parents. The concept of the crossover operator is

applied to a new chromosome with the hope that when it takes the best characteristics from each of the parents, it can produce a better offspring than both parents. There are many types of crossover operators as one-point, two-point, uniform, etc. In the framework of this thesis, I will present one-point, two-point and uniform.

a) One-Point Crossover:

This crossover operates at the individual level. A crossover point is randomly chosen for two individuals (parents) obtained by selection. This point divides each individual into the left and right sections. Then the left (or the right) section of the two individuals is swapped. As an example in Figure 2.3 of crossover, consider the two parents.

Index	1	2	3	4	5	6	7	8	9
Parent 1	1	0	1	0	1	0	1	0	1
Parent 2	1	0	0	0	0	1	0	0	0

Figure 2.3 Parent example.

Suppose the crossover point randomly occurs after the fifth bit. Then all parent bits after the fifth bit are swapped to give the following two children:

Index	1	2	3	4	5	6	7	8	9
Child 1	1	0	1	0	1	1	0	0	0
Child 2	1	0	0	0	0	0	1	0	1

Figure 2.4 New offspring with one-point crossover.

This is a typical crossover-type and used quite commonly in biology.

b) Two- Point Crossover:

The two-point of crossover chosen two randomly points then the bit string is interchanged between two parents from the first random point until the second random point to produce two new offspring. However, the advantage of having more crossover points is that the problem space may be searched more thoroughly. Figure 2.5 show crossover using two points crossover.

Index	1	2	3	4	5	6	7	8	9
Parent 1	1	0	1	0	1	0	1	0	1
Parent 2	1	0	0	0	0	1	0	0	0
Child 1	1	0	0	0	0	1	0	0	1
Child 2	1	0	1	0	1	0	1	0	0

Figure 2.5 Generated next generation using two points crossover at 2nd and 8th.

c) Uniform Crossover:

Index	1	2	3	4	5	6	7	8	9
Parent 1	1	0	1	0	1	0	1	0	1
Parent 2	1	0	0	0	0	1	0	0	0
Child 1	1	0	0	0	1	0	1	0	0
Child 2	1	0	1	0	0	1	0	0	1

Figure 2.6 Generated next generation using uniform crosser.

Uniform crossover considers each bit position of the two parents and swaps them with a specified probability. Suppose the first, third, fourth, and ninth bits positions (of the original parents) are swapped. Figure 2.6 show crossover using uniform crossover.

2.2.4 Mutation

The mutation operator is viewed as a background operator to maintain different genetics in the population. The basic mutation operator generated is the random position of one of the bits in a bit string then the bit corresponding to that position is changed. Finally, copy the generated individual into the new generation of the population. There are many different types of mutation operators used in a binary representation such as flip bit and interchange.

Index	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Parent 1	**1**	**0**	**1**	**0**	**1**	**0**	**1**	**0**	**1**
Parent 2	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
Mutation chromosome	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**0**
Offspring 1	**0**	**1**	**1**	**0**	**0**	**1**	**0**	**0**	**1**
Offspring 2	**0**	**1**	**0**	**0**	**1**	**0**	**1**	**0**	**0**

Figure 2.7 Generated the next generation using flip bit mutation.

The flip bit mutation operator crates the new offspring chromosome based on randomly generated mutation chromosome by changing 0 to 1 and 1 to 0. Figure 2.7 explains the flip bit, mutation operator. Offspring chromosomes are produced by flipping a bit (0 to 1 and 1 to 0) if a mutation chromosome is 1then in parent chromosome, the corresponding bit is flipped.

With interchange mutation, a gene that will be positioned as a permutation to become a new chromosome:

Parent	**1**	**0**	**1**	**0**	**1**	**0**	**1**	**0**	**1**
Offspring	**1**	**0**	**1**	**0**	**1**	**1**	**1**	**0**	**0**

Figure 2.8 Generated the next generation using interchange mutation.

2.3 Term Frequency– Inverse Document Frequency

Term Frequency-Inverse Document Frequency [26] (TF-IDF) is a technique that represents how 'important' a word is to a document in the document set used in data mining. High value can show high importance and it depends on the number of times the word appears in the document but is offset by the frequency of that word in the dataset. It has many uses, most importantly in automated text analysis, and is very useful for scoring words in machine learning algorithms for Natural Language Processing (NLP). A few variants of TF-IDF are often used in search systems as a primary tool for evaluation and arrangement of text based on user queries. TF-IDF is also used to filter those from stopwords in articles such as text summary and text classification.

TF-IDF was invented for document search and information retrieval. It works by increasing proportionally to the number of times a word appears in a document, but is offset by the number of documents that contain the word. So, words that are common in every document such as: "this", "what", "and", "if", etc, rank low even though they may appear many times since they don't mean much to that document in particular. TF-IDF for a word in a document is calculated by multiplying two different metrics which I will present in session 2.3.1 and 2.3.2.

2.3.1 Term Frequency

The term frequency of a word in a document. There are several ways of calculating this frequency, with the simplest being a raw count of instances a word appears in a document. Then, there are ways to adjust the frequency, by the length of a document, or by the raw frequency of the most frequent word in a document. Since the

documents can have different short lengths, some words may appear multiple times in a longer document than a short document. As such, term frequency is usually divided by length document (the total words in a document).

$$\boldsymbol{tf(t,d)} = \frac{\boldsymbol{f(t,d)}}{\boldsymbol{max\{f(t,d): w \in d\}}}$$

where:

- tf(t,d): Term frequency of word t in document d.
- f(t,d): Number of occurrences of word t in document d.
- $max\{f(t,d): w \in d\}$: The number of occurrences of a word with the most number of occurrences in document d.

2.3.2 Inverse Document Frequency

When TF calculations, all the words are treated as having equal importance. But some words like "is", "of" and "that" often appear a lot of times but the importance is not high. So we need to reduce the importance of these words. The inverse document frequency of the word across a set of documents, help evaluate the importance of a word. This means, how common or rare a word is in the entire document set. The closer it is to 0, the more common a word is. This metric can be calculated by taking the total number of documents, dividing it by the number of documents that contain a word, and calculating the logarithm.

So, if the word is very common and appears in many documents, this number will approach 0. Otherwise, it will approach 1.

$$\boldsymbol{idf(t,D)} = \boldsymbol{log}\frac{\boldsymbol{|D|}}{\boldsymbol{|\{d \in D: t \in d\}|}}$$

where:

- Idf(t,D): inverse docume.nt frequency of word t in documents D.
- |D|: total document in D.
- $\{d \in D: t \in d\}$: total document in D include word t.

The logarithm parameter in this formula does not change the IDF value of the word but only narrows the value of the word. Because the change in the baseline will result in the value of the words changed by a certain number and the proportion between the weights together will not change. In other words, changing the baseline will not affect the proportion of IDF values. The use of logarithm to help the TF-IDF value of a smaller word, as we have the TF-IDF calculation of a word in a document in the area of TF and IDF of the word. The higher the score, the more relevant that word is in that particular document. To put it in more formal mathematical terms, the TF-IDF score for the word t in document d from the document set D is calculated as follows:

$$\boldsymbol{tf - idf(t, d, D) = tf(t, d).idf(t, D)}$$

Equation 2-1 Term frequency- inverse document frequency.

2.4 Cosine Similarity

Cosine similarity [27] is a metric used to determine how similar the documents are irrespective of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. The Cosine similarities formula is very simple as in Equation 2-2

$$\boldsymbol{Cos(\theta) = \frac{\overrightarrow{AB}}{|\vec{A}||\vec{B}|}}$$

Equation 2-2 Cosine Similarity.

The meaning of cosine similarity in the context of natural language processing is the semantics of the two words together are reflected through the corners of the vector performing them.

In this Chapter 2, we have learned the basics: What is the automatically summarize, there are 2 types of main text summary methods, genetic algorithms, and its operator. The content in this chapter will be the core platform for us to go to a recommended model for extractive text summarization in Chapter 3.

Chapter 3 Proposed Genetic Algorithm

3.1 Overview

In this chapter, I present the basic steps of the proposed Genetic Algorithm to automatically generate summaries of single documents.

The proposal for this algorithm is to optimize the fitness function based on sentence features, change the approach to initialize, selectively, and mutation against the conventional.

Extractive summarization has output is an optimum solution. Therefore, the use of a genetic algorithm is appropriate. The principle of Genetic Algorithm imitates the process of natural evolution. This process starts with an initial population that is randomly chosen, then creates a new generation from this current one through an iterative way. First, all the individuals (or chromosomes) of the current population are evaluated according to the fitness function. The selection of the best parent is then applied on the basis of these fitness values. Second, the selected individuals are modified through crossover and mutation mechanisms to generate a new "stronger" population. The process of GA is repeated with the new population. This iteration stops until the best solution is reached or until some stop-criteria are raised (e.g., after a number of iterations, resource consumption, etc.).

For text summarization, sentences are clustered and subsequently, the most important sentences from each cluster are selected to form a summary in which each sentence contains sufficiently different information from other sentences in the summary. Hence, the criterion function should achieve two conditions: compactness and separability. Firstly, objects in a given cluster should be very similar to each other (compactness), and secondly, objects in different clusters should be very different from each other (separability).

3.2 General solution

The extractive summarization model uses Genetic Algorithm described in Figure 3.1

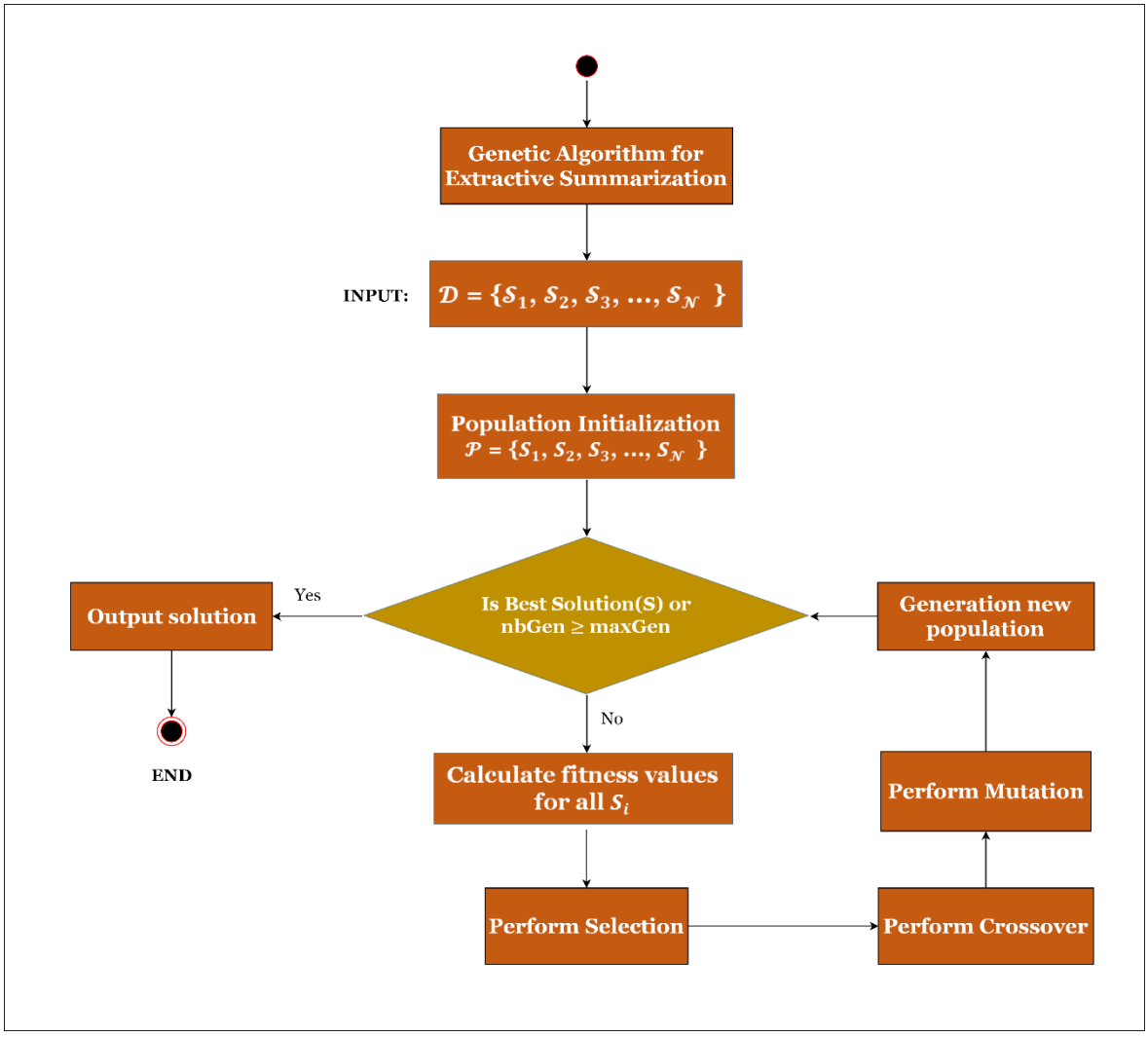

Figure 3.1 The steps of the proposed GA for single documents.

Accordingly, the sequence of steps is as follows:

(i) Read input document $\mathcal{D}$ with $\mathcal{N}$ sentences.

(ii) Initializing the populations with each individual is an assumption summary.

(iii) Check if find the best solution or exceed the number of allowed generations to stop the algorithm, otherwise skip to step (iv).

(iv) Calculating a fitness function for all summaries.

(v) Conduct a selection of good individuals according to the fitnes values calculated in step (iv).

(vi) Crossover the selected good individuals.

(vii) Conducting genetic mutations.

(viii) Create new populations and return to step (iii).

Detailed steps described in section 3.3, 3.4, 3.5 below.

3.3 Document Representation

3.3.1 Document performances

A given document with *N* sentences is represented by the set $\boldsymbol{\mathcal{D}} = \{\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, ..., \mathcal{S}_{\mathcal{N}}\}$ where $\mathcal{S}_i$ corresponds to the *i*-th sentence of this document. In a like manner as Natural Language Processing tasks, the vectorization of document sentences is tackled by using weighting terms. Supposing that there are terms in the document $\boldsymbol{\mathcal{D}}$ after the preprocessing phase to exclude stop words. A sentence is represented by the set $\mathcal{S}_i = \{t_{i1}, t_{i2}, t_{i3}, ... t_{ik}, ... t_{i\mathcal{M}}\}$ where t_{ik} corresponds to the *k*-th term in the *i*-th sentence. The vector representation of $\mathcal{S}_i$, denoted as $\mathbf{S}_i$

$$\mathbf{S}_i = \{w_{i1}, w_{i2}, ..., w_{ik}, ..., w_{i\mathcal{M}}\}$$

where: w_{ik} is the weight of the term t_k in the sentence $\mathcal{S}_i$. This weight is calculated on the basis of term frequency.

The Term Frequency - Inverse Document Frequency (see 2.3) technique is widely used in text mining and information retrieval. In the context of single document summarization, we also use a variation of TFIDF, the *TF-ISF* [4], as follows:

$$\boldsymbol{tf_{i,k}} = \frac{\boldsymbol{f_{i,k}}}{\boldsymbol{MaxFreq_i}}$$

$$\boldsymbol{isf_k} = \boldsymbol{log}\frac{\boldsymbol{\mathcal{N}}}{\boldsymbol{n_k}}$$

where:

- $\mathbb{f}_{i,k}$ is the frequency of the term t_k in the sentence $\mathcal{S}_i$.
- $MaxFreq_i$ is the number of occurrences of the most frequent term in the sentence $\mathcal{S}_i$.
- n_k is the number of sentences in which the term t_k appears and $\boldsymbol{\mathcal{N}}$ is the number of sentence in the document $\boldsymbol{\mathcal{D}}$.

The formulation of the weight w_{ik} is in Equation 3-1:

$$\boldsymbol{w_{ik} = tf_{i,k} . isf_k}$$

Equation 3-1 The formulation of the weight w_{ik}.

Each sentence of the document $\boldsymbol{\mathcal{D}}$ is represented as a weighing vector of all terms extracted from this document. The document $\boldsymbol{\mathcal{D}}$ can be seen as a matrix of $\boldsymbol{\mathcal{N}x\mathcal{M}}$ where $\boldsymbol{\mathcal{M}}$ is the total number of terms of $\boldsymbol{\mathcal{D}}$.

3.3.2 Summary performances solution

The use of GA for solving the extractive text summarization aims to search all the sentences of the document and fnd those maximizing the fitness function. In literature, a Genetic Algorithm solution is typically represented using a binary vector [14, 15, 21]. Thus, if a document $\boldsymbol{\mathcal{D}}$ has $\boldsymbol{\mathcal{N}}$ sentences, the candidate summary to GA is formed as a vector of $\boldsymbol{\mathcal{N}}$ elements with 0 or 1 values, for example **S** = {0, 1, 0, 1, ..., 0, 1} where 0*s* mean that these sentences are not in the summary and 1*s* indicates that the summary contains the corresponding sentences. In that sense, a long document would be represented by a large and sparse vector. In this thesis, I propose an alternative way to represent the solution vector in which only the index of *in-summary* sentences are stored. For example, given a document $\boldsymbol{\mathcal{D}}$ of $\boldsymbol{\mathcal{N}}$ = 25 sentences, the length of summary (in *sentences*) is $\boldsymbol{\ell}$ = 5, the solution vector, instead of a sparse vector of 25 elements, is **S** = {1, 2, 5, 9, 10} indicating that the summary is composed of the first, second, fifth, ninth and tenth sentences from the source document.

3.4 Proposed of fitness function

3.4.1 Sentence features for proposed GA

In this thesis, I select some commonly used features to evaluate the quality of a summary. These are a similarity to the topic sentence, sentence length, sentence position, number of proper nouns.

3.4.1.1 Similarity to the topic sentence

Topic sentences are sentences that carry the general meaning plays an important role in the document. Topic sentences are usually placed at the beginning of every text. Calculating the similarity of sentences with topic sentences to find sentences with similar content to the topic in the text, thereby creating a full informative summary that the paragraph provides grant. All the sentences of the summary should be related to the topic. The cosine similarity between two sentences is calculated based on Equation 2-2 as follows.

$$\boldsymbol{similarity_{cos}(Si, Sj)} = \frac{\sum_{k=1}^{\mathcal{M}}(w_{ik} \cdot w_{jk})}{\sqrt{\sum_{k=1}^{\mathcal{M}}(w_{ik}^2 \, x \sum_{k=1}^{\mathcal{M}}(w_{jk}^2)}}$$

where:

- $\mathcal{M}$ refers to the total number of terms in the document.
- w_{ik} denotes the weight of the term *k* in the sentence $\mathcal{S}_j$.
- w_{jk} denotes the weight of the term *k* in the sentence $\mathcal{S}_j$.

The similarity to the topic of all sentences in the summary **S** is then calculated as in Equation 3-2.

$$\mathcal{R}_S = \frac{\sum_{\forall \mathcal{S}_i \in S} \boldsymbol{similarity_{cos}(S_i S_q)}}{\ell}$$

where:

- $\mathcal{S}_q$ is the topic sentence.

- $\boldsymbol{\ell}$ is the length of the summary $\mathbf{S}$ (measured in sentences).

Equation 3-2 The similarity to the topic of all sentences in the summary.

The normalized relation of sentences with the topics is be defined as in Equation 3-3.

$$\widehat{\mathcal{R}}_S = \frac{\mathcal{R}_S}{max_{\forall s_i \in S} \mathcal{R}_i}$$

where:

- max $\mathcal{R}_i$ denotes the sentence of the summary having the highest similarity to the topic.
- $\widehat{\mathcal{R}}_S$ characterizes the relation of the summary $\mathbf{S}$ with the topic sentence.

Equation 3-3 The normalized relation of sentences with the topics.

The similarity feature $\widehat{\mathcal{R}}_S$ characterizes the relation of the summary $\mathbf{S}$ with the topic sentence, whose value is between 0 and 1. The more $\widehat{\mathcal{R}}_S$ is closed to 1, the more the content of the summary S is related to the topic and vice versa.

3.4.1.2 Sentence length

The shortest sentences (e.g., exclamatory sentences) are less likely to appear in the summary. Therefore, the length of all sentences in the summary is taken into account to qualify this characteristic. The length (measured by words) of all sentences in the summary is calculated and then normalized:

$$\hat{\boldsymbol{\ell}}_i = \frac{\boldsymbol{\ell}(\boldsymbol{s}_i) - \boldsymbol{\mu}}{\boldsymbol{\sigma}}$$

where: μ and σ are the average length and standard deviation of all the sentences in the summary respectively.

The sentence length factor of the summary $\mathbf{S}$ is defined as in Equation 3-4.

$$\mathcal{L}_S = \frac{\sum_{\forall s_i \in S} \hat{\boldsymbol{\ell}}_i}{max_{\forall s_i \in S} \hat{\boldsymbol{\ell}}_i}$$

where: $\max\hat{\ell}_i$ denotes the length of the longest sentence in the summary.

Equation 3-4 The sentence length factor of the summary.

3.4.1.3 Sentence position

As presented, the most important information of the document tends to appear in specific sections such as titles, headings or the opening of paragraphs, etc. Therefore, evaluating the summary based on the position of its sentence allows taking into account the distance between them and the key sentences of the document. The position feature of the summary **S** is evaluated as in Equation 3-5.

$$\mathcal{P}_S = \frac{\sum_{\forall S_i \in S} \mathcal{N} - Pos(S_i)}{\mathcal{N}}$$

where:

- $Pos(i)$ refers to the position of the sentence S_i (of the summary **S**) in the original document $\mathcal{D}$
- $\mathcal{N}$ denotes the number of sentences of the document $\mathcal{D}$.

Equation 3-5 The position feature of the summary.

In this way, $\mathcal{P}_S$ tends to favor the summaries that contain sentences belonging to the first sentences of the document.

3.4.1.4 Number of proper nouns

This feature gives more relevance to the sentences which contain proper nouns, under the idea that proper nouns tend to relate to relevant information [14]. The evaluation of the proper noun factor of a summary can be seen in Equation 3-6.

$$\mathcal{N}_S = \frac{\sum_{\forall S_i \forall S} \mathcal{N}_{S_i}}{\mathcal{N}_{\mathcal{D}}}$$

where:

- $\mathcal{N}_{S_i}$ represents the number of proper nouns of the sentence S_i.

- $\mathcal{N}_{\mathcal{D}}$ refers to the total number of proper nouns of the document $\mathcal{D}$.

Equation 3-6 The evaluation of the proper noun factor of a summary.

In this equation, $\mathcal{N}_{S}$ has high values when sentences in summary contain more proper nouns and vice versa.

3.4.2 Fitness function

As mentioned above, the objective of the fitness function is to assess each individual (or chromosome) of the current population. According to fitness values, the "strongest" parents are selected to create the next generation. The fitness function used in this work is based on the features introduced in 3.2. Given a candidate summary $\boldsymbol{S} = \{\boldsymbol{x_i}\}_{\boldsymbol{i=1}}^{\boldsymbol{\ell}}$ where x_i is the corresponding index of the sentence $\boldsymbol{S_i}$ in the original document $\mathcal{D}$, $\boldsymbol{\ell}$ is the number of sentences of the summary. The quality of **S** is assessed by maximizing the fitness function $\mathcal{f}$**(S)** according to Equation 3-7.

$$\mathcal{f}(\boldsymbol{S}) = \boldsymbol{\alpha}\widehat{\mathcal{R}}_{\boldsymbol{S}} + \boldsymbol{\beta}\mathcal{L}_{\boldsymbol{S}} + \boldsymbol{\gamma}\mathcal{P}_{\boldsymbol{S}} + \boldsymbol{\sigma}\mathcal{N}_{\boldsymbol{S}}$$

where: $\alpha, \beta, \gamma, \sigma$ are coefficients which give a weighting to each featur

Equation 3-7 Fitness function for summarization.

$\alpha, \beta, \gamma, \sigma$ parameters satisfy Equation 3-8.

$$\boldsymbol{\alpha + \beta + \gamma + \sigma = 1}$$

Equation 3-8 Sum of weighting to each feature.

3.5 Proposed strategies for genetic operators

3.5.1 Population initialization

At the beginning of the algorithm, some candidates are selected to create the first generation. The initial selection is performed randomly, under the idea that all the sentences of the document should have an equal probability of appearing in the summary. To achieve that, a random number from 1 to is taken, where $\mathcal{N}$ is the

number sentences of the original document $\mathcal{D}$. Under the hypothesis that each candidate summary has a pre-fixed length (in sentences) ℓ and the population size is *n*. In GA terminology, the initial population is then a set of *n* chromosomes whose gens are sentences' index:

$$\mathcal{P}(0) = \{\mathcal{S}_i(0)\}_{i=1}^{n}$$

With

$$\mathcal{S}_i(0) = \{x_{i,j}(0)\}_{j=1}^{l}$$

where: $x_{i,j}$ has the value in [1..$\mathcal{N}$] corresponding to the index of the selected sentence.

3.5.2 Parent Selection

At each iteration of the algorithm, some better individuals (i.e., summaries) are selected from the current population according to their fitness values. There are many selection strategies such as *Roulette Wheel* selection, *Tournament* selection, *Rank* selection etc. [19]. In this study, I adopt the *Roulette Wheel* selection for parent selection. After applying the fitness function to all individuals, a probability will be assigned to each by the following formula:

$$p_i = \frac{f_i}{\sum_{i=1}^{n} f_i}$$

where: f_i is the fitness value of the *i*-th individual from the population $\mathcal{P}$.

To be selected, the cumulative probability of the *i*-th individual p_{cum_i} should exceed a random value in the range [0..1]

$$p_{cum_i} = \sum_{j=1}^{i} p_j$$

In this manner, *m* individuals will be selected for the next generation where *m* is a pre-fixed parameter ($m < n$).

3.5.3 Crossover

To generate new offspring, in this research, I adopt two crossover strategies: *one-point* and *uniform* crossover. The desired goal is to compare these mechanisms to explore which one is more appropriate than the other in terms of accuracy of the generated summary as well as the time to convergence of GA.

a) One-point crossover

The main idea of this strategy is to pick randomly a point on both parents' chromosomes to exchange their parts. Given two parents chromosomes S_f and S_m (with the character f stands for *"father"* and m for *"mother"*), the new offspring S_c is defined as below:

$$\boldsymbol{S_c[i]} = \begin{cases} \boldsymbol{S_f[i]} & \boldsymbol{if\ (i \leq pt)} \\ \boldsymbol{S_m[i]} & \boldsymbol{otherwise} \end{cases}$$

where:

- S[i] denotes the i-th element of the vector **S.**
- $\boldsymbol{pt}$ is a random value in the range [1..$\boldsymbol{\ell}$].
- $\boldsymbol{\ell}$ is the size (in sentences) of the summary.

Equation 3-9 Children with one-point crossover.

The other offspring is created by swapping the role of the parents and following the same process.

b) Uniform crossover

This strategy considers that each gene from either parent has an equal probability of being chosen. Given a random mixing ratio a ($a \in [0..1]$) and two parents chromosomes $\boldsymbol{S_f}$, $\boldsymbol{S_m}$, the new offspring is built as following.

$$\boldsymbol{S_c[i]} = \begin{cases} \boldsymbol{S_f[i]} & \boldsymbol{if\ (p_i \geq a)} \\ \boldsymbol{S_m[i]} & \boldsymbol{otherwise} \end{cases}$$

where: $\boldsymbol{p_i}$ is randomly picked between 0 and 1 for each gene.

Equation 3-10 Children with uniform crossover.

The generation of the second offspring is carried out by exchanging the role of the parents with the same value of $\boldsymbol{p_i}$. Compared with previous works, in our study, as the solution vector $\mathbf{S}$ contains only the index of *in-summary* sentences, there is no need to consider the summary length (in sentences) constraint after crossover, thus leading to a gain in execution time of GA.

3.5.4 Mutation

The mutation aims to alter one or more genes of a chromosome, replaces their values by other information. In the context of extractive summarization, this operator is typically performed by choosing randomly a bit and flipping its value (i.e., from 0 to 1 or vice versa). In that sense, the summary length constraint should be also considered. In this study, two mutation strategies are investigated: a *random* strategy and a *guided* strategy.

a) Random mutation strategy

There are two common approaches for random mutating. The first approach only considers the genes with value 1 for flipping. In contrast, the second only considers the genes with value 0. I adopt the former for random mutation. Each gene of a chromosome has an equal probability to be mutated. Its value will be replaced by an arbitrary value *x* in the range [1... $\mathcal{N}$] where $\mathcal{N}$ is the number of sentences of the original document $\mathcal{D}$. Before mutating, it is necessary to check whether the *x*-th sentence has appeared in the summary or not. Another value of *x* would be chosen in case of conflict.

$$\boldsymbol{S_{new}[i]} = \begin{cases} \boldsymbol{round(p_i \mathcal{N})} & \boldsymbol{if\ (p_i \geq a_{mut}} \\ \boldsymbol{S[i]} & \boldsymbol{otherwise} \end{cases}$$

where:

- $\boldsymbol{a_{mut}}$ is the mutation probability
- $\boldsymbol{p_i}$ is a random real value in the range [0..1]

- $\boldsymbol{x_i} = \boldsymbol{round}(\boldsymbol{p_i}\mathcal{N})$ is the new value to assign to the *i*-th gene.

Equation 3-11 Mutation with random strategy.

b) Guided mutation strategy

Another mutation strategy is proposed in this work, under the idea that the relevant information of the document tends to be found in the opening sentences of paragraphs. Thus, I mutate a randomly selected gene by replacing its value by the index of signifcant sentences (i.e., leading sentences of paragraphs). I designate this strategy as "guided" mutation strategy

$$\boldsymbol{S_{new}[i]} = \begin{cases} \boldsymbol{x^*} & \boldsymbol{if\ (p_i \geq a_{mut})} \\ \boldsymbol{S[i]} & \boldsymbol{otherwise} \end{cases}$$

where:

- x^* is the index of some leading sentences of paragraphs in the original document $\boldsymbol{\mathcal{D}}$.
- $\boldsymbol{a_{mut}}$ is the mutation probability.
- $\boldsymbol{p_i}$ is a random real value in the range [0..1].

Equation 3-12 Mutation with guided strategy.

In a like manner as the random mutation strategy, the selected gene is only mutated if the replaced value is not in the representation vector.

3.6 Related works

Genetic Algorithm was used in many research disciplines as a solution for global search optimization [2, 5]. In the context of text summarization, GA is cast as a clustering method allowing to find the most relevant sentences for the summary. The most important component of GA is the fitness function. This function helps evaluate candidates to select the best ones. From the point of view of the unsupervised learning tasks, the fitness function produces a ranking of sentences. The evaluation is often performed using some text attributes. Many studies have been realized to investigate

such attributes. Silla et al. proposed to use GA for the selection of text attributes in order to improve the performance of supervised classification algorithms applying in automatic text summarizing [19]. The authors experimented with different compositions of attributes such as the position of sentence, length of sentence, average TF-ISF (a variation of Term Frequency– Inverse Document Frequency measure), similarity to the title, similarity to keywords, etc. They evaluated the accuracy of the classifier (i.e., decision tree-induction and Naïve Bayes) to explore the effectiveness of their GA-based selection attribute. In this work, all the text features have the same importance. Different from the work of Silla et al. which only used GA for selecting text attributes before applying supervised learning methods in classifying sentences, Meena et al. proposed to use GA to adjust weights associated with text features [14]. Their goal was to improve the fitness function after each iteration of the algorithm by optimizing the weight value associated to each text feature. Each sentence will be assigned a score given by the fitness function. The summary is built by selecting the highest score sentences. In both the works [14] and [19], it is not possible to know which features are more relevant. This is because there are not restricted weight values considered for all the features. While the aforementioned works focus on exploring the feature space using GA (In the GA terminology, the search space is the text features), some other studies considered the text summarization task as finding/selecting the best candidates among all the sentences of the original document to compose good summaries (i.e., maximizing the fitness function) [4, 7, 15, 21]. Following these approaches, an individual or chromosome is often encoded using binary representation. Each gene of the chromosome, representing a sentence in the document, has the value of 0 if it is considered in the summary or 1 if inversely. The fitness function is also built on the basis of text features but is calculated for all the sentences from the candidate summary. Consequently, the quality of summaries is considered during the exploration of search space. In the GA terminology, the operators including selection, crossover as well as mutation help finding the best summary which maximizes the fitness function. To the best of our knowledge, all the operators are typically performed at a random level. In this thesis, I propose to use GA to solve the same

problem. Inspired from the previous works, I build the fitness function on the basis of some features such as the similarity to the topic, sentence length, sentence position, number of proper nouns. The main objective of our work is to investigate the relevance of those features and instead of random operators, to propose a “guided” mutation/crossover mechanism in order to compose qualified summaries and improve the performance of GA.

Chapter 4 Experimental analysis

In this section, the evaluation and comparison of proposed method with other related works are presented. The evaluation of this research uses a set of metrics n-gram statistics. This study chooses ROUGE-1, ROUGE-2 and ROUGE-L as the measurement for the experimental results using the average precision, recall and f-measur. I also explain rouge measurement and the evaluation was performed by comparing the generated summaries with the labelled references.

4.1 Evaluation metrics

4.1.1 Precision, recall and f-score

In this thesis, I used the method of calculating accuracy through sentences for extractive text summary. The evaluation was performed by comparing the generated summaries with the labeled references. This means I will use recall and precision at the sentence level instead of the word level. I use three measurements of recall, precision, and F-score, which are presented in section 4.1.1.1, 4.1.1.2, 4.1.1.3.

4.1.1.1 Recall

To get a good quantitative value, we can actually compute the precision and recall using the overlap. Simply put, recall refers to how much of the reference summary the system summary is recovering or capturing. It can be computed as:

$$Recall = \frac{number\ of\ overlapping\ words}{total\ words\ in\ reference\ summary}$$

4.1.1.2 Precision

A machine-generated summary (system summary) can be extremely long, capturing all words in the reference summary. But, many of the words in the system summary may be useless, making the summary unnecessarily verbose. So we can use precision

to avoid this. The amount of precision used to evaluate the summary text by the system how many pieces of information in which meaningful. Precision is measured as:

$$Precision = \frac{number\ of\ overlapping\ words}{total\ words\ in\ system\ summary}$$

4.1.1.3 F-score

We have F-measure is used to balance system performance on both Precision and Recall measures. F-Score can be considered as the harmonic mean of Recall and Precision. A text summary system with a greater number of F-measure means the more efficient the system summary.

$$F - measure = \frac{2.Recall.Precision}{Recall + Precision}$$

4.1.2 Rouge measure

The ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is a system for measuring the quality of summaries by comparing it to summaries are created by humans, ROUGE is proposed by (Lin and Hovy 2003, Lin 2004a) [24]. The measure is computed by counting the number of overlapping words or n-grams between the system-generated summary to be evaluated and the ideal summaries. ROUGE-N, ROUGE-S, and ROUGE-L can be thought of as the granularity of texts being compared between the system summaries and reference summaries. For example, **ROUGE-1** refers to overlap of ***unigrams*** between the system summary and reference summary. **ROUGE-2** refers to the overlap of ***bigrams*** between the system and reference summaries. This thesis using ROUGE-1, ROUGE-2, and ROUGE-L. I will present about them in sections 4.1.2.1, 4.1.2.2, 4.1.2.3.

4.1.2.1 Rouge 1

As mentioned above, ROUGE-1 refers to the overlap of 1-gram (each word) between the system and reference summaries. For clarity, let us see the following example:

System summary:

the woman who wears a red jacket is my teacher

Reference summary:

my teacher wears a red jacket

So, we have:

System summary unigrams									
the	woman	who	wears	a	red	jacket	is	my	teacher
Reference summary unigrams									
my	teacher	wears	a	red	jacket				

Based on the unigrams above, the ROUGE-1 recall is as follows:

$$ROUGE - 1_{recall} = \frac{6}{6} = 1$$

This means that all the words in the reference summary have been captured by the system summary, which indeed is the case for this example.

Now the ROUGE-1 precision is as follows:

$$ROUGE - 1_{precision} = \frac{6}{10} = 0.6$$

This simply means that 6 out of the 10 words in the system summary were, in fact, relevant or needed.

4.1.2.2 Rouge 2

ROUGE-2 refers to the overlap of bigrams (2 words) between the system and reference summaries. With the example above, we have:

System summary bigrams								
the woman	woman who	who wears	wears a	a red	red jacket	jacket is	is my	my teacher
Reference summary bigrams								
my teacher	teacher wears	wears a	a red	red jacket				

The ROUGE-2 recall is as follows:

$$ROUGE - 2_{recall} = \frac{4}{5} = 0.8$$

Similar, the ROUGE-2 precision is as follows:

$$ROUGE - 2_{precision} = \frac{4}{9} = 0.44$$

4.1.2.3 Rouge L

Rouge L measures the longest matching sequence of words using the Longest Common Subsequence (LCS) based statistics. An advantage of using LCS is that it does not require consecutive matches but in-sequence matches that reflect sentence-level word order. Since it automatically includes the longest in-sequence common n-grams, we don't need a predefined n-gram length.

In this example, we have "***wears a red jacket***" is longest common subsequence. The ROUGE-L recall is as follows:

$$ROUGE - L_{recall} = \frac{4}{6} = 0.67$$

The ROUGE-L precision is as follows:

$$ROUGE - L_{precision} = \frac{4}{10} = 0.4$$

4.2 Preparation for experiment

4.2.1 Datasets

For my experiments, I have used the CNN/DailyMail corpus which is originally constructed by Hermann et al. [9], statistics the total number of documents for each dataset that is described in Table 4.1

Table 4.1 *Daily Mail/CNN corpus description*

Datasets	**CNN**	**DailyMail**
Train	90,266	196,961
Validation	1,220	12,148
Test	1,093	10,397

Each document is associated with a gold summary (i.e., abstractive summary) created by human experts and an extractive summary in which all sentences were labeled: 0 - not in the summary; 1 - absolutely in the summary and 2 – might be in the summary. On average, there are about 28 sentences per document and about 3 to 4 sentences per reference gold summary and about 11 sentences per reference label summary.

In this study, I used the CNN corpus for tuning the fitness function weighing parameters (i.e., α, β, γ, σ) in Equation 3-8. This corpus was also used for evaluating the effectiveness of GA operators. In order to assess the quality of generated summaries and compare with the other related works, the DailyMail corpus was used.

4.2.2 Environment setup

To build a text summary model using genetic algorithms, I use the following programming tools:

- For the preprocessing, I deploy in Python language 3.7 [28] and the programming library for handling natural language NLTK [29] and deployed on jupyter notebook with computer configuration core i3-4030U.
- For experimental process, I also deploy in Python language 3.7 and using VGA GTX 1080 Ti, CPU core i7.
- For summarunner model, I re-implemented SummaRunner model in Pytorch language and using CPU core i7 to training model.
- For textrank, I use the library Text Rank [30] for python and proceed with the installation of the data set presented.

The Daily Mail dataset includes original and references, where on average each gold reference file has about 3 to 4 sentences per reference gold summary and about 11 sentences per reference label summary. Therefore, I chose the summary length parameter (in sentences) $\boldsymbol{\ell}$ = 4 for model comparison with the supervised learning method and $\boldsymbol{\ell}$ = 11 for model comparison with unsupervised.

The population size was set to n = 1000 individuals for each iteration of GA. At the parent selection phase, I kept 65% of the population for creating the next generation (i.e., m = 650). The random mixing ratio (see Equation 3-10) for performing uniform crossover was set to a = 0.5 and the mutation probability (see Equation 3-11 and Equation 3-12) was set to a_{mut} = 0.4.

The evaluation metric used in this research was conducted by means of ROUGE measures in section 4.1.2. Furthermore, I also adopted the precision/recall measures for automatic extractive summarization evaluation.

4.3 Experiment for fitness function

Before applying GA, the original documents were pre-processed by first removing all stop words and non-alphanumeric symbols. The parameter tuning process was carried out on the CNN corpus and divided into two stages.

In the first stage, I conducted a selection of sentence features, researched and appreciated their importance for the automatic summarization. As presented in section 3.4.1, I experimented and comparative sentence features:

- Topic Similarity (TS): The normalized relation of sentences with the topics.
- Sentence Length (SL): The sentence length factor of the summary.
- Sentence Position (SP): The position feature of the summary
- Number of Proper Nouns (NPN): The prope noun factor of a summary.

The results, as can be seen in Table 4.2, show that the most relevant features are Sentence Length and Number of proper nouns, next up is position the sentences in the document as well as the similarities with the topic sentence. However, not all the best results together will produce the best results, or weak results together will produce poor results. Results are calculated in ROUGE measures on generated summaries with label references.

Table 4.2 *ROUGE (F1) Scores of the proposed GA with the fitness function based on each sentence feature and its combination*

Method	**ROUGE-1 (%)**	**ROUGE-2 (%)**	**ROUGE-L (%)**	**Convergent Speed (s)**
GA+TS	54.2	54.9	61.3	1.27
GA+SL	**67.8**	**58.2**	**64.7**	**1.41**
GA+NPN	**68.6**	**59.7**	**65.9**	**1.39**
GA+SP	64.3	55.2	61.5	1.73
GA+NPN+SL	69,7	61,0	67,1	1.40
GA+TS+SL+NPN+SP	**70,4**	**62,3**	**68,2**	**1.43**
GA+TS+SL	69,8	61,1	67,3	1.41

Regarding the results of the first state, in the second one, I combined all the features and experimented some sets of values for four weights (α, β, γ, σ) which correspond to the features of Similarity to the topic sentence($\widehat{\boldsymbol{\mathcal{R}}}_{\boldsymbol{S}}$), Sentence length ($\boldsymbol{\mathcal{L}}_{\boldsymbol{S}}$), Sentence position ($\boldsymbol{\mathcal{P}}_{\boldsymbol{S}}$) and Number of proper nouns ($\boldsymbol{\mathcal{N}}_{\boldsymbol{S}}$), respectively. For each experiment, different values of relevance to each feature were tested. I found that the weights obtaining good results are: **α = 0.25, β = 0.3, γ = 0.15, σ = 0.3.**

4.4 Experiment for GA operators strategies

In this section, the effectiveness of different initialization, crossover and mutation strategies is evaluated according to the ROUGE measures on generated summaries with label references. I have conducted experiments several times by using different ways.

4.4.1 Initialization

First, I proceeded to randomly initialize and initialize with good chromosomes from the process of ranking sentence evaluation using the TextRank Algorithm.

Table 4.3 *ROUGE (F1) Scores of the proposed GA with different initialization strategies*

Method	**ROUGE-1 (%)**	**ROUGE-2 (%)**	**ROUGLE-L (%)**	**Convergent Speed (s)**
Randomly initialization	**68.8**	**59.6**	**66.0**	**1.23**
Initialization with textrank algorithm	68.5	59.5	65.9	2.46

As a result of Table 4.3, the natural initialization process produces better results than initialized with the good individuals selected from the sentence ranking of the Text Rank algorithm. However, it is hard to explain this because nature has so many unexpected things that humans cannot explain. In fact, the normal couple somehow

was born a genius child with outstanding features. In this case, although Rouge does not rise too much, the performance is significantly reduced due to the saving time of the algorithm. So, I choose to randomly initialize to save initialization time for my model.

4.4.2 Crossover

Second, I compared two crossover strategies: one-point crossover and uniform crossover strategies. As figured out in Table 4.4, the ROUGE evaluations are insignificantly different, the uniform strategy gives better convergent speed than the one-point strategy. This result is very feasible for documents of larger lengths.

Table 4.4 *ROUGE (F1) Scores of the proposed GA with different crossover strategies*

Method	**ROUGE-1 (%)**	**ROUGE-2 (%)**	**ROUGLE-L (%)**	**Convergent Speed (s)**
One- point	70.2	62.0	68.0	1.98
Uniform	**70.4**	**62.3**	**68.2**	**1.03**

4.4.3 Mutation

Finally, I studied the performance of the two mutation mechanisms: random mutation and guided mutation mechanisms. Table 4.5 depicts that the proposed GA with guided mutation mechanism converges more quickly than the random one. Although this is difficult to demonstrate in theory, however, I have experimented on CNN's dataset with over 2000 documents.

Thus, I choose the uniform crossover and guided mutation mechanisms for conducting the next experiment.

Table 4.5 *ROUGE (F1) Scores of the proposed GA with deifferent mutation strategies*

Method	**ROUGE-1 (%)**	**ROUGE-2 (%)**	**ROUGLE-L (%)**	**Convergent Speed (s)**
Random mutation	70.2	62.0	67.8	1.33
Guided mutation	**70.4**	**62.3**	**68.2**	**1.02**

4.5 Experimental result

In this research, I compared our proposed model with another unsupervised extractive summarization, the TextRank model [16] as well as a supervised extractive summarization, the SummaRunNer model [17].

The Text Rank model was constructed by Rada Mihalcea which adopted the graph based ranking algorithm for scoring sentences. The summaries were built by choosing the best scored sentences.

The SummaRunNer model, proposed by Nallapati et al., is a recurrent neural network based sequence model for extractive summarization. The authors used two recurrent neural networks to extract representation features of documents at the word level. The sentence features then were extracted from the word-level by using another two recurrent neural networks. As a supervised extractive summarization, the training phase should be performed on a large document corpus (i.e., DailyMail/CNN).

I re-implemented these two aforementioned models and performed the experiment on 1000 test documents of the DailyMail corpus. I calculated the ROUGE measures by comparing the generated summaries with the associated gold reference.

Table 4.6 *ROUGE (F1) Scores of the proposed GA comparing with Text Rank and SummaRunNer. The experiment was carried out on 1000 test documents of Daily Mail corpus. The evaluation was performed by comparing the generated summaries with the associated gold references*

Method	**ROUGE-1 (%)**	**ROUGE-2 (%)**	**ROUGLE-L (%)**
Proposed GA	37.8	15.5	26.4
Text Rank (unsupervised)	36.7	11.8	21.7
SummaRunNer (supervised)	38.1	15.7	21.4

Because the gold summary used to compare to the abstractive method should be in this case the result and the summary are not high. However, this comparison aims to synchronize reference with SummaRunNer.

As can be observed in Table 4.6, our proposed GA gives greater ROUGE values than Text Rank (increasing 1.1% ROUGE-1 and 3.7% ROUGE-2 but slightly lower than SummaRunNer (decreasing 0.3% ROUGE-1 and 0.2% ROUGE-2). However, as a deep learning extractive summarization model, a large training corpus with all the labeled sentences is required in the case of SummaRunNer. Our proposed model, in contrast, does not ask for any training data. In that sense, our model achieves a better performance in terms of time execution and resource consumption.

I then evaluated the effectiveness of all three models by comparing the generated summaries with the extractive references. All the sentences labeled by 1 are classified in the summary. All the others (with label 0 and 2) are not in the summary. As the comparison was performed sentence by sentence, in this experiment, the other metrics were considered: precision, recall, and F-measure (4.1.1).

Table 4.7 *Precision and Recall Scores of the proposed GA comparing with Text Rank and SummaRunNer. The experiment was carried out on 1000 test documents of DailyMail corpus. The evaluation was performed by comparing the generated summaries with the labeled references*

Method	**F-measure (%)**	**Precision (%)**	**Recall (%)**
Proposed GA	**58.0**	**84.3**	**48.3**
Text Rank (unsupervised)	50.8	47.5	54.6
SummaRunNer (supervised)	51.1	86.6	39.0

The results in Table 4.7 show that our proposed model outperformed these two other ones, i.e., increasing 6.9% and 7.2% on F-measure comparing with SummaRunNer and Text Rank, respectively.

The example below is a summary obtained from GA. Highlights sentence are sentences in the labeled reference. The results showed that were 9/11 sentences accurate compared to the reference have 9 sentences.

Original Document Original document example presented in appendix C
Summary by GA with 4 sentences Leeds assistant manager Steve Thompson (left) has been suspended by the club Steve Thompson's suspension has left manager Neil Redfearn considering his future at Elland Road a club statement read: The director of football Salerno, has today (april 2) suspended the assistant coach Steve Thompson from his duties at the club. Neil Redfearn is considering his position as manager of Leeds after the club suspended his assistant Steve Thompson on thursday.

Leeds took two months to appoint Steve Thompson but he was told in a letter from director of football Salerno yesterday morning that he was be suspended until the end of the season effectively ending his time at Elland Road after 105 days.

Steve Thompson is understood to have been told that he will be suspended until his contract expires at the end of the season

Gold Summary by Human

Steve Thompson suspended by leeds just 19 games in to no 2 role. Speculation mounting over future of manager neil redfearn. Leeds are 13th in the championship with 52 points.

Summary by GA with 11 sentences

Neil Redfearn is considering his position as manager of Leeds after the club suspended his assistant Steve Thompson on thursday.

Leeds took two months to appoint Steve Thompson but he was told in a letter from director of football Salerno yesterday morning that he was be suspended until the end of the season effectively ending his time at Elland Road after 105 days.

Steve Thompson is understood to have been told that he will be suspended until his contract expires at the end of the season.

He has also been informed that an option to renew his deal will not be taken up by the club.

Leeds assistant manager Steve Thompson (left) has been suspended by the club Steve Thompson's suspension has left manager Neil Redfearn considering his future at Elland Road a club statement read: The director of football Salerno, has today (april 2) suspended the assistant coach Steve Thompson from his duties at the club.

This is an internal matter and the club will make no further comment on this internal issue.

Neil Redfearn, the club's third coach of the season, took training before attending a delayed press conference and said: It’s become difficult for me.

I’ve got to have a good think about my future now.

We have put things together me and Steve Thompson.

Neil Redfearn’s contract is also up at the end of the season and club president Massimo Cellino is suspended until may 3.

Sol Bamba heads home in a recent win for the Whites at Fulham's Craven Cottage Leeds sit 13th in the Championship table but were unbeaten in march and take on Blackburn Rovers on Saturday.

Referece Document with labeled sentences

Neil Redfearn is considering his position as manager of Leeds after the club suspended his assistant Steve Thompson on thursday.

Leeds took two months to appoint Steve Thompson but he was told in a letter from director of football Salerno yesterday morning that he was be suspended until the end of the season effectively ending his time at Elland Road after 105 days.

Steve Thompson is understood to have been told that he will be suspended until his contract expires at the end of the season.

Leeds assistant manager Steve Thompson (left) has been suspended by the club Steve Thompson's suspension has left manager Neil Redfearn considering his future at Elland Road a club statement read : The director of football Salerno , has today (april 2) suspended the assistant coach Steve Thompson from his duties at the club.

Neil Redfearn, the club's third coach of the season, took training before attending a delayed press conference and said: It’s become difficult for me.

We have put things together me and Steve Thompson.

Neil Redfearn’s contract is also up at the end of the season and club president Massimo Cellino is suspended until may 3.

Sol Bamba heads home in a recent win for the Whites at Fulham's Craven Cottage Leeds sit 13th in the Championship table but were unbeaten in march and take on Blackburn Rovers on Saturday.

Chapter 5 Conclusion and future works

5.1 Conclusion

Research has given me the opportunity to learn about machine learning and machine learning approaches. As for text summaries, researchers have previously proposed approaches to supervised and unattended learning.

In this thesis, I proposed a GA based method for extractive text summarization. I focused on improving the fitness function by evaluating the relevance of sentence features. Moreover, I improved the basic crossover and mutation mechanisms to increase the accuracy of generated summaries. I performed extensive experiments with different scenarios using DailyMail/CNN corpus to study the effectiveness of the proposed methods. The empirical results show that the proposed GA gives better accuracy in comparison with TextRank and SummaRunNer, increasing the accuracy by 7.2% and 6.9% respectively. The experimental results showed that our model outperformed the state of the arts.

The solution of Enhanced Genetic Algorithm for Single document Extractive Summarization presented in this thesis has a paper accepted at SOICT 2019 [1] (The 10th International Symposium on Information and Communication Technology).

5.2 Future works

In this model, the weight of each sentence feature is very important. I am currently using experiments many times. In the future, I hope to be able to use machine learning to find the weight of each feature for each document.

Currently, I use the CNN/Daily mail DataSet for my model. I would also like the future to create or find a set of Vietnamese language data sets to summarize the

Vietnamese document. Moreover, I also want to apply GA for multi-document summarization.

Each sentence feature has a certain degree of influence. In recent studies have shown 3 features groups with over 15 categories. I hope I can find out and improve the new sentence features that is more suitable for text summaries.

The reality indicates that nature always contains marvelous things. I look forward to researching and finding a natural algorithm that fits the text summary and delivers better results and performance than genetic algorithm.

References

[1] B. T. M. Anh, N. T. My, and N. T. T. Trang, “Enhanced Genetic Algorithm for Single Document Extractive Summarization,” p. 7, 2019.

[2] Pratibha Bajpai and Manoj Kumar. 2010. Genetic algorithm–an approach to solve global optimization problems. *Indian Journal of computer science and engineering* 1, 3 (2010), 199–206.

[3] Michael Buckland and Fredric Gey. 1994. The relationship between recall and precision. *Journal of the American society for information science* 45, 1 (1994), 12–19.

[4] Niladri Chatterjee, Amol Mittal, and Shubham Goyal. 2012. Single document extractive text summarization using genetic algorithms. In *2012 Third International Conference on Emerging Applications of Information Technology*. IEEE, 19–23.

[5] Kalyanmoy Deb. 1998. Genetic algorithm in search and optimization: the technique and applications. In *Proceedings of international workshop on soft computing and intelligent systems*. Machine Intelligence Unit, Indian Statistical Institute Calcutta, India, 58–87.

[6] Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artifcial intelligence research* 22 (2004), 457–479.

[7] René Arnulfo García-Hernández and Yulia Ledeneva. 2013. Single extractive text summarization based on a genetic algorithm. In *Mexican Conference on Pattern Recognition*. Springer, 374–383.

[8] David E Goldberg. 2006. *Genetic algorithms*. Pearson Education India.

[9] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems*. 1693–1701.

[10] Ju-Hong Lee, Sun Park, Chan-Min Ahn, and Daeho Kim. 2009. Automatic generic document summarization based on non-negative matrix factorization. *Information Processing & Management* 45, 1 (2009), 20–34.

[11] Wenjie Li, Mingli Wu, Qin Lu, Wei Xu, and Chunfa Yuan. 2006. Extractive summarization using inter-and intra-event relevance. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 369–376.

[12] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*. 74–81.

[13] Chin-Yew Lin and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *Proceedings of the 18th conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, 495–501.

[14] Yogesh Kumar Meena and Dinesh Gopalani. 2015. Evolutionary algorithms for extractive automatic text summarization. *Procedia Computer Science* 48 (2015), 244–249.

[15] Martha Mendoza, Susana Bonilla, Clara Noguera, Carlos Cobos, and Elizabeth León. 2014. Extractive single-document summarization based on genetic operators and guided local search. *Expert Systems with Applications* 41, 9 (2014), 4158–4169.

[16] Rada Mihalcea. 2004. Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*. 170–173.

[17] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. SummaRuNNer: A Recurrent Neural Network Based Sequence Model for Extractive Summarization of Documents.. In *AAAI*. 3075–3081.

[18] Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685* (2015).

[19] Carlos N Silla, Gisele L Pappa, Alex A Freitas, and Celso AA Kaestner. 2004. Automatic text summarization with genetic algorithm-based attribute selection. In *Ibero-American Conference on Artifcial Intelligence*. Springer, 305–314.

[20] Scott M Thede. 2004. An introduction to genetic algorithms. *Journal of Computing Sciences in Colleges* 20, 1 (2004), 115–123.

[21] Eder Vázquez, Rene Arnulfo Garcia-Hernandez, and Yulia Ledeneva. 2018. Sentence features relevance for extractive text summarization using genetic algorithms. *Journal of Intelligent & Fuzzy Systems* 35, 1 (2018), 353–365.

[22] Kam-Fai Wong, Mingli Wu, and Wenjie Li. 2008. Extractive summarization using supervised and semi-supervised learning. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*. Association for Computational Linguistics, 985–992.

[23] Jen-Yuan Yeh, Hao-Ren Ke, Wei-Pang Yang, and I-Heng Meng. 2005. Text summarization using a trainable summarizer and latent semantic analysis. *Information processing & management* 41, 1 (2005), 75–95.

[24] K. Ganesan, “ROUGE 2.0: Updated and Improved Measures for Evaluation of Summarization Tasks,” *Computational Linguistics*, vol. 1, no. 1, p. 8.

[25] R. Mihalcea and P. Tarau, “TextRank: Bringing Order into Text,” in *Proceedings of EMNLP 2004*, Barcelona, Spain, 2004, pp. 404–411.

[26] Robertson, S. (2004), "Understanding inverse document frequency: on theoretical arguments for IDF", Journal of Documentation, Vol. 60 No. 5, pp. 503-520.

[27] “Cosine Similarity - an overview | ScienceDirect Topics.” [Online]. Available: https://www.sciencedirect.com/topics/computer-science/cosine-similarity. [Accessed: 26-Dec-2019].

[28] “PyPI: The Python Package Index,” *PyPI*. [Online]. Available: https://pypi.org/. [Accessed: 26-Dec-2019].

[29]“Natural Language Toolkit — NLTK 3.4.5 documentation.” [Online]. Available: https://www.nltk.org/. [Accessed: 26-Dec-2019].

[30] P. X. Nathan, *pytextrank: Python implementation of TextRank for phrase extraction and summarization of text documents*.

Appendix

A Sentence Features for Extractive Summarization

Recently, various sentence features were proposed for extractive summarization. First, I will present a detailed description of the four types of sentence features, i.e., surface, content, event, and relevance features, which will be examined systematically.

A.1 Surface Features

Surface features are based on the structure of documents or sentences. It includes sentence position [13] in the document, the number of words in the sentence [14], and the number of quoted words in the sentence. Table 0.1 indicates the description of these types of features.

Table 0.1 *Types of surface features*

Sentence feature	**Description**
Sentence position	1/sentence no.
Doc_first	Whether it is the first sentence of a document
Para_first	Whether it is the first sentence of a paragraph
Sentence length	The number of words in a sentence
Quocte	The number of quoted words in a sentence

The intuition with respect to the importance of a sentence stems from the following observations:

(1) The first sentence in a document or a paragraph is often a topic sentence, which carries the content and significance representing it.

(2) The sentences in the earlier parts of a document are more important than sentences in later parts;

(3) Stop words are the words that bring additional meaning, not important and affect the document. So, a sentence is important if the number of words (except stop words) in it is within a certain range;

(4) Quotes give additional meaning to the content to mention. Therefore, a sentence containing too many quoted words is unimportant.

A.2 Content Features

In terms of content, sentence features are based on content-bearing words i.e., centroid words, signature terms, and high-frequency words. Both unigram and bigram representations have been investigated. Table 0.2 description of the six content features.

Table 0.2 *Types of content features*

Sentence feature	**Description**
Centroid_Uni	The sum of the weights of centroid uni-gram
Centroid_Bi	The sum of the weights of centroid bi-grams
SigTerm_Uni	The number of signature unigrams
SigTerm_Bi	The number of signature bi-grams
FreqWord_Uni	The sum of the weights of frequent uni-grams
FreqWord_Bi	The sum of the weights of frequent bi-grams

A.3 Event Features

An event is comprised of an event term and associated event elements. An occurrence of an event term (or event element) in a document is considered as an instance, while the collection of the same event terms is considered as a concept. Given a document set, instances of event terms and event elements are identified first. An event map is then built based on event instances or concepts. PageRank algorithm is used to assign a weight to each node (an instance or concept) in the event map. The final weight of a sentence is the sum of weights of event instances contained in the sentence.

A.4 Relevance features

Relevance features are incorporated to exploit inter-sentence relationships:

(1) sentences related to important sentences are important.

(2) sentences related to many other sentences are important.

Relevance features are incorporated to exploit inter-sentence relationships:

(1) sentences related to important sentences are important.

(2) sentences related to many other sentences are important.

The first sentence in a document or a paragraph is important, and other sentences in a document are compared with the leading ones. Two types of sentence relevance, FirstRel_Doc, and FirstRel_Para (see Table 0.3), are measured by comparing pairs of sentences using word-based cosine similarity (section 2.4). Another way to exploit sentence relevance is building a sentence map. Every two sentences are regarded as relevant if their similarity is above a threshold. Every two relevant sentences are connected with a unidirectional link. Based on this map, the PageRank algorithm is applied to evaluate the importance of a sentence.

Table 0.3 *Types of relevance features*

Sentence feature	Description
FirstRel_Doc	Similarity with the first sentence in the document
FirstRel_Para	Similarity with the first sentence in the paragraph
PageRankRel	PageRank value of the sentence based on the sentence map

B Text Rank algorithm for extractive textsumarization

B.1 Introduction

Text Rank is the summary method of extracting text. This model is proposed based on the Page Rank algorithm and graph-based algorithms. This algorithm is proposed in 2004 and is highly appreciated in the method of extractive text summarization. [25]

The Page Rank algorithm is an algorithm that analyzes the links on the Internet that is used by Google in the search and arrangement of links there are important levels in descending order. We can understand an important link is a link that is directed by other important links. So this idea is recursive. The Page Rank value of a link in the probability for the user to access that link.

B.2 Theoretical basis

Based on the idea of Page Rank, Text Rank believes that a sentence is highly informative when it is related to other high-value information sentences. Is that a recursive idea? And the relevance between these verses can be calculated based on the number of words that appear in those sentences or calculated based on other similar personality-like attributes such as Cosine similarities.

B.3 Text Rank model

Text Rank uses graph structure to show and calculate sentence rank in text. The tops of the graphs are sentences, the edges between the vertices are weighted between 2

sentences. The weight here represents the similarities between the two verses represented by the two peaks. As mentioned, the similarities between the two sentences can be calculated based on the number of words that appear in either two sentences or use the same formula as Cosine. The steps taken in the Text Rank algorithm are described as in Figure 0.1.

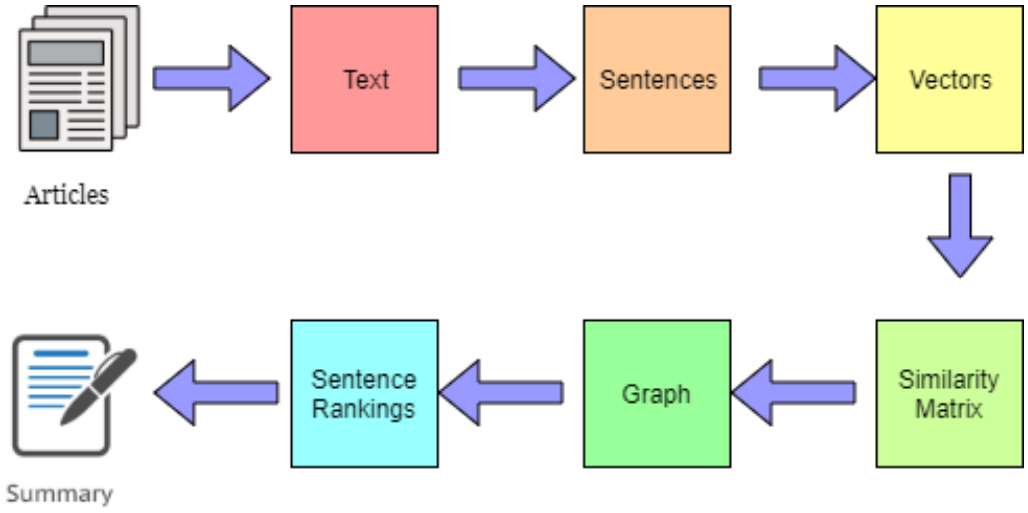

Figure 0.1 Text Rank algorithm diagram.

In Figure 0.1 we see the general idea for extraction algorithms that are always ranking sentences and picking out high-ranking sentences. Step 1 and Step 2 will be two steps for pretreatment. Step 1 separates the sentences in text and the sequence numbers to make it more convenient for later pairing. In the 2nd step, we'll segment each sentence into words, or phrases. Besides, it is possible to remove words called "Stop word". The words "Stop word" are often words that appear frequently in text and do not bear a high value of information.

Step 3 calculates the similarities between sentences. In the graph, the contrast will be performed at the edge between the two peaks of the graph.

Step 4, we will create the graph to store the sentence. In essence, this is just a programming technique to store the graphs of sentences for the computer can handle.

Step 5 to calculate the weighted average of each sentence has a lot of formula variations for the Text Rank. These formulas are based on the weight of the related

sentence to calculate the average weighted of the current sentence. After a weighted sentence, we will rank them in descending order.

And the last step is to pick out the highest weighted sentences in a defined ratio and align folded in the original order to get a text summary.

As such we see for the Text Rank algorithm, the weight of this peak will depend on other peaks and recursively. The formula of the Text Rank has different variations and will result in different improvements. Besides, the Text Rank algorithm is highly appreciated when it can carry comparisons with state of the art extraction methods.

The effective Text Rank is because it not only works at the local level (i.e. works only on each vertex of the graph), which works across the graph. Moreover, the Text Rank reviews each peak is very important thanks to the perspective of delegating it to other important peaks (which are performed by the formula recursively weighted each peak). In addition, through the mechanism of review each peak, the weight of the sentence is gradually more accurate.

C Original Document example

"Neil Redfearn is considering his position as manager of Leeds after the club suspended his assistant Steve Thompson on Thursday. Leeds took two months to appoint Steve Thompson but he was told in a letter from director of football Salerno yesterday morning that he was be suspended until the end of the season effectively ending his time at Elland Road after 105 days. Steve Thompson is understood to have been told that he will be suspended until his contract expires at the end of the season. He has also been informed that an option to renew his deal will not be taken up by the club. Leeds assistant manager Steve Thompson (left) has been suspended by the club Steve Thompson's suspension has left manager Neil Redfearn considering his future at Elland Road a club statement read: the director of football Salerno, has today (april 2) suspended the assistant coach Steve Thompson from his duties at the club. This is an internal matter and the club will make no further comment on this internal issue. Neil Redfearn, the club's third coach of the season, took training

before attending a delayed press conference and said: It's become difficult for me. It's not ideal timing. I've got to have a good think about my future now. We have put things together me and Steve Thompson. Before it was a no brainer for me to stay on, not so much now. I love this club, I was brought up on Leeds but this situation is a difficult one for me. Thommo was informed this morning through a letter. He's been suspended - for what, I don't know. You'd have to find that out from the club. I don't understand why he has been suspended. It's a difficult one for me to take, I can't think for one minute why someone who has been part of this success has been sacked. Me and Steve Thompson were the ideal partnership, he's been great for me and I'm bitterly disappointed that he's not here. Neil Redfearn's contract is also up at the end of the season and club president Massimo Cellino is suspended until may 3. Sol Bamba heads home in a recent win for the Whites at Fulham's Craven Cottage Leeds sit 13th in the Championship table but were unbeaten in march and take on Blackburn Rovers on saturday. It is also understood that Neil Redfearn is under pressure not to select top scorer Mirco Antenucci who will trigger a bonus clause in his contract should he score two more goals. Steve Thompson is taking advice from the LMA while Salerno's own position is thought to be open to review at the end of the season."