Republic of Iraq
Ministry of Higher Education and Scientific Research
University of Babylon
College of Information Technology
Software Department

Unified Objective Ant Colony Optimization for Sentiment Oriented Text Summarization

A Thesis
Submitted to the Council of the College of Information Technology at University of Babylon in Partial Fulfillment of the Requirements for the Degree of Master in Information Technology/Software

By

Abeer Raad Hassan Ali

Supervised by

Assoc. Prof. Dr. Rafid Sagban Abbood Kareem

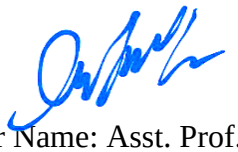
2022 A.C. **1444 A.H**

﴿ قَالُواْ سُبۡحَٰنَكَ لَا عِلۡمَ لَنَآ إِلَّا مَا عَلَّمۡتَنَآۖ
إِنَّكَ أَنتَ ٱلۡعَلِيمُ ٱلۡحَكِيمُ ﴾

سورة البقرة: اية (٣٢)

Supervisor Certification

I certify that the thesis entitled (**Unified Objective Ant Colony Optimization for Sentiment Oriented Text Summarization**) was prepared under my supervision at the department of Software/ College of Information Technology/ University of Babylon, by **Abeer Raad Hassan** as partial fulfillment of the requirements of the degree of Master in Information Technology-Software.

Signature: 

Supervisor Name: Asst. Prof. Dr. Rafid Sagban Abbood

Date: / /2023

The Head of the Department Certification

In view of the available recommendations, I forward the thesis entitled (**Unified Objective Ant Colony Optimization for Sentiment Oriented Text Summarization**) for debate by the examination committee.

Signature:

Prof. Dr. Ahmed Saleem Abbas

Head of Software Department

Date: / /2023

Certification of the Examination Committee

We hereby certify that we have studied the dissertation entitled (**Unified Objective Ant Colony Optimization for Sentiment Oriented Text Summarization**) presented by the student (**Abeer Raad Hassan**) and examined him/her in its content and what is related to it, and that, in our opinion, it is adequate with (Viva Result) standing as a thesis for the degree of Master in Information Technology-Software.

Signature:
Name: Dr. Asaad Sabah Hadi
Title: Professor
Date: / / 2022
(Chairman)

Signature:
Name: Dr. Mohammed Hasan Abdulameer
Title: Assistant Professor
Date: / / 2022
(Member)

Signature:
Name: Ahmed Khelfa Al-Ajeli
Title: Assistant Professor
Date: / / 2022
(Member)

Signature:
Name: Dr. Rafid Sagban Abbood
Title: Assistant Professor
Date: / / 2022
(Member and Supervisor)

Approved by the Dean of the College of Information Technology, University of Babylon.

Signature:
Name: Dr. Hussein Atiyah Lafta
Title: Professor
Date: / / 2022
(Dean of Collage of Information Technology)

Declaration

I hereby declare that this thesis, submitted to University of Babylon in partial fulfillment of requirement for the degree of Master in Information Technology \ Software, has not been submitted as an exercise for a similar degree at any other University. I also certify that this work described here is entirely my own except for experts and summaries whose source are appropriately cited in the references.

Signature:
Name: Abeer Raad Hassan
Date: / /2023

Dedication

To the owner of first and last gratitude, to the guide to the right path

God Almighty

To the one who delivered the message and fulfilled the trust...and advised the nation...to the prophet of mercy and the light of the worlds

Our Master Muhammad, may God bless him and grant him peace

To those who wrote with their blood the most wonderful pages of glory, redemption, sacrifice and giving

The martyrs of Iraq

To my angels who lift me up when my wings forget how to fly

My family

To those who linked me to the relationship of lineage, the fragrance of friendship and the roses of love

Acknowledgements

In the beginning, thanks for the One who does not need these words because He is closer to me than myself, and He knows the extent of my recognition, my thanks, and my gratitude for the One who supports me, and without Him, this task would not have completed. Thanks for the One who has never left me alone, thank because you were and still my support and never let me down, thank you, my Lord.

I would like to thank my supervisor **Dr. Rafid Sagban** for his countless help, support, guidance, and knowledge he provided me with throughout my research.

To the inhalation and exhalation, which I cannot live without them… my parents... Thank you for every minute of your life that you sacrificed for me.

To the constant and supportive love for me … my beloved husband (Ali).... I thank you for your constant support and help, without you, I could not have done anything.

To my eyes and my heart…My children (Abdullah and Roya).... Thank you for your patience. I am very proud of you, even though I have been busy with my studies, you scored the highest grades and are ahead of your peers.

To my inexhaustible inheritance…my brothers and sisters (Dina, Yasmin, Sabah, Muhammad, and Mustafa).... Thank you for your love, for your support, for believing in me.

Last but not least, I would like to thank all the kind, helpful and lovely people who helped me directly or indirectly to complete this work and apologize to them for not being able to mention them by name here, but they are in my heart.

Abeer Raad

Declaration Associated with this Thesis

Some of the works presented in this thesis have been published as below.

First Paper:

Title: Optimization-Based Techniques for Sentiment-Oriented Text Summarization: A Concise Review

Author: Abeer Raad and Rafid Sagban,

Journal: NeuroQuantology,

Volume: 20,

Number: 8,

Pages: 2230-2238,

Year: 2022,

Publisher: Anka Publisher.

Second Paper:

Title: A Unified Objective Ant Colony Optimization for Sentiment Oriented Text Summarization

Author: Abeer Raad and Rafid Sagban,

Conference: International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT),

Pages: 182-188,

Year: 2022,

Publisher: IEEE Xplore.

Abstract

In automatic text summarization, computer algorithms are used to automatically produce a meaningful and informative summary of the input text. To immediately analyse public opinion about any event, sentiment analysis and opinion mining require summarization techniques. The most commonly used approaches to solve this problem is by using various optimization methods, which is interested in finding the best solution under certain constraints. These optimization algorithms will use the quality of summarization as an objective function. However, in this thesis, a Unified Objective Ant Colony Optimization for Sentiment Oriented (SO.ACO) algorithm is used in a unique way. Instead of feeding the documents directly to the algorithm to find the best summarization, this proposed approach considers the sentiment score and the TF-IDF similarity as a distance matrix for the Ant Colony Optimization. The graph of the Ant Colony Optimization content nodes and edges, each node represents a sentence and each edge represent the distance between two sentences. The most connected sentence with the other sentences (the sentence in the middle of the document) is considered the most important sentence. Instead of relying only on the distance as an objective function of the Ant Colony Optimization algorithm, three criteria are considered for this aim: sentiment relevance, content coverage, and redundancy reduction. Also, it is used to upgrade the heuristic function which is the second base of solution construction in the ACO.

The dataset used in this work is DUC 2002, which has 509 documents divided into 59 topics. All of them are news related texts and the supported summarization is carried out manually by experts.

The metrics used in the evaluation of the proposed approach are the Rouge-1, Rouge-2, and Rouge-L, where the sentiment relevance, content coverage, and redundancy reductions are calculated between the generated summaries and the manually summaries. Furthermore, the Pearson correlation is calculated between the algorithm generated and manually generated summarization. The obtained results are as follows: Rouge-1: 0.601, Rouge-2: 0.5294, Rouge-L: 0.5908; the sentiment relevance: 1.868, content coverage: 0.78, and redundancy reductions: 0.627. In addition, the Pearson correlation coefficient is 0.8603. These results are higher than the results of the existing approaches, where the average percentage improvement was as follows: Rouge-1 : + 20.23 % and Rouge-2 : + 207.21 %.

Table of Contents

Chapter Three: The Proposed System

Chapter Four: The Experimental Results and Discussion

Chapter Five: Conclusions and Future Work

List of Table

List of Figures

List of Algorithms

List of Abbreviation

Abbreviation	Meaning
ACO	Ant Colony Optimization
AI	Artificial Intelligence
ATS	Automatic Text Summary
CO	Combinatorial Optimization
CSA	Crow Search Algorithm
DT	Decision Tree
ESA	Explicit Semantic Analysis
IE	Information Extraction
IR	Information Retrieval
LSA	Latent Semantic Analysis
MMR	Maximal Marginal Relevance
NLP	Natural Language Processing
RNN	Recurrent Neural Network
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
SC	Soft Computing
Seg2Seg	sequence to sequence
SEO	Search Engine Optimization
SRL	Semantic Role Labelling
SS	Sentiment Analysis
SVD	Singular Value Decomposition
SVM	Support Vector Machine
TF-IDF	Term Frequency-Inverse Document Frequency
TSP	travel salesman problem
VADER	Valence Aware Dictionary and sEntiment Reasoner

Chapter One

General Introduction

General Introduction

1.1 Introduction

Nowadays with the emerge of information technologies in every aspect of life (also known as the fourth industrial revolution), there is a flood of data that surrounds people every day. This emergence led to the abundance of information and easy access to it. However, this abundance comes with a time cost. Since there is a high volume of data available online (mostly text data), there is little time available to read all the desired texts.

This leads to the importance of automatic text summarization techniques. This requires a method to summarize the important parts of the text, and rank them according to their importance and set a threshold to select the most important sentences. This results in present in different applications, such as creating summarized newsletters, Search Engine Optimization (SEO), legal contract analysis, media monitoring, and others[1][2].

There are two main types of text summarization: are extractive and abstractive. In the first one the most important sentences are selected from the original text[3]. While in the second one new text that holds the most important sentences is generated from the original data[4] .The following two procedures are used to choose summary sentences. The first is the greedy selection method, which selects the best textual units (e.g., sentences) one by one. This method is popular for text summarizing since it is simple and quick; yet, it rarely delivers the best results[5]. The second is the global optimal selection strategy, looks for the summarization with the highest quality, rather than the summarization with the greatest sentences. It turns the work of summarizing, or at least the step of picking sentences, into an optimization problem[6], in which the overall score of

the output summary is improved by finding the optimal combination of sentences. Several summarizing objectives, including as content coverage, the redundancy reduction, and the sentiment relevance, have been investigated and optimized in the literature. Thus, since there are more than one (and usually conflicting) criteria to be considered, the automatic text summarization are usually classified as Multi-Objective Optimization Problem. The objectives produce a summary covering the main content, reducing the redundancy, and reflecting the sentiment of the authors' opinions. To solve this optimization problem, meta-heuristics algorithms applied such as genetic algorithm, ant colony optimization and particle swarm optimization[7]. The criteria of the content coverage, the redundancy reduction, and the sentiment relevance have been defined as objective functions.

Practically, text summarization problem is formulated as an optimization problem, which should be modeled as a similarity graph. Ant colony optimization (ACO) is a prominent meta-heuristic in which any optimization problem should be model as a construction graph to enable a multi agent called artificial ant to search this graph looking for on optimal solution to the optimization problem being considered[8][9]. This matching between the similarity graph and construction graph motives the application of the ACO.

In (ACO) finding optimal path in a graph depend on two criteria first, meta-heuristic to get high similarity between two sentence, second, amount pheromone deposited by the ant where choose paths that have strong pheromone to find best path. In short, the match between the ACO model and the text summarization model is a solution to the computational problems that can be reduced to finding good paths (sentences) in the

graphs, for this the ACO was used, as it proved effective and relatively low cost approach to solve difficult optimization issues.

1.2 Problem statement

In contrast to other automatic text summarization methods, sentiment-oriented is a highly relevant approach to the polarity of user opinions contained in texts. There are three criteria that should be considered in the design of automatic text summarizers: covering the main content, reducing the redundancy, and reflecting the sentiment of the authors' opinions. The problem of sentiment-oriented extractive multi-document text summarization entails the optimization of the three criteria, specifically, multi-objective optimization. ACO is a prominent search method where the solution construction, heuristic and pheromone update functions are the core of its search strategy. However, the existing ACO-based summarizers suffer from several shortcomings. Firstly, they do not consider the matching between the model of the algorithm and the model of the optimization problem where the construction graph of the ACO matches the similarity graph of text summarization. Secondly, the amount of deposit pheromone is only derived from the quality of best so far solutions with no utilization of the three criteria (i.e. content, redundancy and the authors' opinion) in the pheromone update component. Thirdly, they do not consider any information useful for designing suitable heuristic functions. It is expected that application of such improvements to the ACO framework will contribute to the solvers of sentiment-oriented text summarization problems.

1.3 Research Objectives

The primary purpose of this research is to introduce an Ant Colony model and assess its capacity to summarize text effectively.

Following are the study's precise objectives:

1. To formulate a new heuristic function that is usable for in construction solution in the ACO.
2. To develop an ACO algorithm that unify the three objective functions in a single objective function.
3. To evaluate the quality and effectiveness of an extracted summary based on the proposed approach.

1.4 Research questions

The purpose of this research is to provide answers to the questions listed below:

1. How can the new heuristic function contribute to giving the best solutions (best summary) for the ACO algorithm?
2. How to unify the function of the three objectives of the search into one objective function in the environment of the algorithm of Ant Colony Optimization (ACO)?
3. Does the proposed approach enhance the performance of optimization-based automatic text summarization methods?

1.5 Research significance

Automatic text summarization is critical in many fields, as it can be used as a web service on different applications such as summarizing articles and news, summarizing legal documents and summarizing sentiment analysis etc. The purpose of sentiment's orientation is to determine opinionated sentences with their orientation in which the sentiment score is calculated for each input text.

This research attempts to improve the accuracy of the results in automatic text summarization, as the total results exceeded the previous literature.

1.6 Related work

This section displays the works that have already been done on text summarization utilizing distinctive methods. There are different techniques which are connected for text summarization some of them related to the current research as presented in the following lines:

In 2021, Sanchez-Gomez et al. [10] proposed a method for sentiment-oriented text summarization that is query-based. This method achieves two stages. The first is unified objectives (query relevance, sentiment relevance, and redundancy reduction). The second stage is to apply the Crow Search Algorithm (CSA) to optimize the sentence selection. The used dataset is TAC2008. The documents are preprocessed, and each word is given a score. Then it is supplied to the CSA to find the best solution. The Rouge metric is used to evaluate the model. The results are as follows: Rouge-1: 0.4728, Rouge-2: 0.2987.

In 2020, Sanchez-Gomez et al.[11], proposed a method to summarize documents using extractive approach. The proposed method is based on two things, first the usage of unified objectives. In which more than one objective is provided to the optimization algorithm. Second is the usage of Bee Colony Optimization algorithm. The proposed method is tested on Document Understanding Conferences (DUC 2002) dataset. The results of this work considered only the documents with the top 10 accuracies in the datasets. With average values of Rouge-1: 0.553, Rouge-2: 0.342, and Rouge-L: 0.562.

In 2020, Gunel et al. [12], proposed a method for abstractive summarization that is based on encoder-decoder transformer model. In which they used the Wikidata articles as entity-level knowledge. Considering the structural nature of the data and the world. And for the encoder-decoder transformer, they developed a transformer that is inspired by the Transformer-XL decomposition. CNN/Daily Mail dataset is used. And the results are Rouge-1: 0.33804, Rouge-2: 0.12509, and Rouge-L: 0.31225.

In 2020, Li et al.[13], also used encoder-decoder based model for abstractive summarization system. They used the Soft attention approach in the decoding phase to get semantic and contextual data about the documents. Furthermore, a double attention pointer network is combined with the Soft attention approach to generate better summarization results. Additionally, Reinforcement Learning besides scheduled sampling are both used to increase the model performance. CNN/Daily Mail and the LCSTS datasets are both used. And the results are Rouge-1: 0.4072, Rouge-2: 0.1828, and Rouge-L: 0.3735 for CNN/Daily Mail dataset. And Rouge-1: 0.4152, Rouge-2: 0.2728, and Rouge-L: 0.3705 for LCSTS dataset.

In 2019, Derek Miller[14] proposed a method that is based on Bidirectional Encoder Representations from Transformers (BERT) model, and K-means clustering to calculate the sentences that are closer to the centre of the document. Their model is deployed on web service (API). The goal is to provide the students a summarization of their lectures notes according to the number entered by the student. According to the author, since there is no standard metric or dataset to measure the quality of lectures summarization, the human examination is used to evaluate the proposed system. One of the weaknesses of the system its poor summery

when small percentage of sentences are required for the document. Another weakness of the system relies in its selection of some words that needs ahead explanation or mention.

In 2019, Lucky and Girsang [15] proposed a Multi-Objective Ant Colony Optimization system. In which the sentences in each document is constructed in similarity matrix that is based on cosine similarity between the sentences, and the word count (i.e. the less the words number, the higher the metric). The similarity matrix is indirect graph. The authors chooses the social media content to apply their system. They choose Twitter tweets, and filtered only the tweets related to the election in Indonesia. Which are 4856 tweets totally. After 500 iterations and 10 trails runs of MOACO. The results are evaluated by comparing the performance of the proposed system with TextRank, LexRank, SumBasic, Latent Semantic Analysis, and KL-Sum. The metrics used in comparing are information reduction and size reduction. The proposed system performance was the best one in the size reduction metric, and the second one in information reduction metric (LexRank was the best one).

In 2019, Elbarougy et al. [16], proposed a system to summarize Arabic language documents. The documents first represented as graph. In which each node is a sentence. And the sentences are connected together with edges that represent the cosine similarity between the sentences. Three phases are used in this system: pre-processing, feature extraction, and finally the graph construction. After these three parts, modified PageRank is applied to select the most relevant sentences. EASC Corpus is used as the dataset for this project. And the best obtained results are as follows. Precision: 0.6875, Recall: 0.7294, and F-measure: 0.6799.

In 2019, Alguliyev et al. [17], proposed a two stages system to perform text summarization. These two stages are based on clustering and

optimization. The first stage is to use k-means clustering to detect the topics in the documents. The second stage is to use optimization to detect the coverage and diversity of the summarization. This is provided with the ability to control the length of the sentences selected. This system performance is compared with other 14 systems to examine its performance. DUC 2001, and DUC 2002 datasets are used to examine the model. For DUC 2001, the Rouge-1 is 0.4727 (the third best model comparing with the other 14), and the Rouge-2 is 0.2012 (the second best one). And for DUC 2002, the Rouge-1 is 0.4908 (the best one), and for Rouge-2 is 0.2309 (the best one too).

In 2019, Abdi et al.[18] proposed a method for text summarization that is considered sentiment-oriented. Two stages are integrated to produce the summary. The first stage is sentiment analysis, which uses several strategies to detect and overcome some drawbacks. The second stage is sentiment summarization. It uses a ranking model based on a graph that integrates linguistic and statistical methods with sentiment information to optimize the result of sentence ranking. This method consider to redundancy and content coverage. DUC 2002 datasets are used to examine the model. And the result are Rouge-1 is 0.4781 and Rouge-2 is 0.0869.

In 2018, Al-Saleh and Menai[19] proposed a method for text summarization following the extractive approach. Ant Colony Optimization (ACO) is used as the optimization algorithm. Documents are provided to the ACO in order to select the highest content coverage of the resulted sentences. The used dataset is Text Analysis Conference 2011 MultiLing Pilot, which consist of English and Arabic documents. The documents are pre-processed and each word is given a score. And then supplied to the ACO to find the best solution. The results for the English

documents are Rouge-1: 0.47397, Rouge-2: 0.17737, and Rouge-L: 0.440136. and for the Arabic documents are Rouge-1: 0.3118, Rouge-2: 0.12019, and Rouge-L: 0.284108.

In 2018, Vazquez et al. [20] proposed a genetic algorithm to optimize the selection of features relevant for extractive text summarization. They take care of some measures to summarize the text from them, sentence position, sentence length, coverage, similarity with the title, and term length feature. The results showed the superiority of the proposed method over previous studies, as well as the relevance of the features to the problem shown. The DUC2002 dataset is used to evaluate this method by Rouge metric. Rouge-1 is 0.48423 and Rouge-2 is 0.22471.

In 2017, Mosa et al.[21] , used the Ant Colony Optimization with the approach of Jensen-Shannon Divergence, to summarize the comments of social media. In this system, cyclic graph is constructed with two criteria's in mind. The first one is the isolation of the long texts. And the second is the similarity of the sentences are used to construct the graph. This graph is optimized by using ACO. The used dataset is a collection of the Facebook posts and their comments. The best obtained results are as follows. Precision: 94 .7% , Recall: 88.9% and F-measure: 91 .7%.

In 2015, the work of Hassan thesis[22] , is developed by using Ant Colony Optimization (ACO) for extractive text summarization. DUC 2002 dataset is used to evaluate the work. The evaluation is done by using ROUGE metric. The obtained results for ROUGE-1 result is 0.516, and ROUGE-2 is 0.284.

A Summary of the related work with additional information are listed in Table (1.1):

Table (1.1) Summary of the related work

Ref. No	Method name	Summary type	Dataset	Evaluation measure	criteria	model match	Result	
[10]	Crow search algorithm	Extractive	TAC 2008	Rouge	Query relevance, redundancy reduction , sentiment relevance	No	Rouge-1: 0.4728 Rouge-2: 0.2987	
[11]	Artificial Bee Colony	Extractive	DUC 2002	Rouge	Content coverage, redundancy reduction	No	Rouge-1: 0.553 Rouge-2: 0.342 Rouge-L: 0.562	
[12]	Encoder-decoder transformer model	Abstractive	CNN/ Daily Mail	Rouge	cohesion the Content	No	Rouge-1: 0.33804 Rouge-2: 0.12509 Rouge-L: 0.31225	
[13]	Soft attention approach & a double attention pointer network	Abstractive	CNN/ Daily Mail & LCSTS	Rouge	cohesion the Content	No	CNN/Daily Rouge-1: 0.4072 Rouge-2: 0.1828 Rouge-L: 0.3735	LCSTS Rouge-1: 0.4152 Rouge-2: 0.2728 Rouge-L: 0.3705
[14]	BERT model with K-means clustering	Extractive	Transcripts derived from Udacity	Human examination	Content coverage, redundancy reduction	No	The result was no perfect but it provided the next step in quality when compared to dated approaches.	
[15]	a Multi-Objective Ant Colony Optimization	Extractive	Social media comments	Cosine Distance & Words Count	Content coverage, redundancy reduction	Yes	Cosine Distance: 0.127 Word count: 388	
[16]	PageRank	Extractive	EASC Corpus	Precision , Recall , F-measure	Content coverage, redundancy reduction	No	Precision: 0.6875 Recall: 0.7294 F-measure: 0.6799	
[17]	k-means with differential evolution	Extractive	DUC 2001 & DUC 2002	Rouge	Content coverage, redundancy reduction	No	DUC2001 Rouge-1: 0.4727 Rouge-2: 0.2012	DUC2002 Rouge-1:0.490 Rouge-2:0.230
[18]	Soft computing	Extractive	DUC2002	Rouge	Content coverage, redundancy reduction	No	Rouge-1: 0.4781 Rouge-2: 0.0869	

[19]	Ant Colony Optimization	Extractive	TAC 2011 MultiLing Pilot (English and Arabic)	Rouge	Content coverage, redundancy reduction	No	English Rouge-1: 0.47397 Rouge-2: 0.17737 Rouge-L: 0.44013	Arabic Rouge-1: 0.311 Rouge-2: 0.120 Rouge-L: 0.284
[20]	Genetic algorithm	Extractive	DUC2002	Rouge	Content coverage, redundancy reduction	No	Rouge-1: 0.48423 Rouge-2: 0.22471	
[21]	Ant Colony Optimization with Jensen-Shannon Divergence	Extractive	Facebook posts and their comments	Precision , Recall , F-measure	Content coverage, redundancy reduction	Yes	Precision: 94 .7% Recall: 88.9% F-measure: 91 .7%	
[22]	Ant colony Optimization	Extractive	DUC2002	Rouge	Content coverage, redundancy reduction	Yes	Rouge-1 Precision:0.516 Recall: 0.51642 F-measure:0.5162	Rouge-2 Precision:0.28416 Recall: 0.24026 F-measure: 0.256

1.7 Thesis structure

Additionally to chapter one, four chapters will be presented, these are:

Chapter Two: reviews an extensive description of the main concepts of text summarization as well as ant colony optimization algorithm which are used in this thesis.

Chapter Three: presents the proposed system, and practical stages for the suggested system with algorithm.

Chapter Four: clarifies the implementation of the proposed methodology in the DUC 2002 dataset and experimental results of this implementation are obtained.

Chapter Five: draws conclusions that have been reached through this thesis and gives suggestions for future work.

Chapter Two

Theoretical Background

Theoretical Background

2.1 Introduction

The purpose of this chapter is to provide an overview of the fundamentals of text summarization, its types, text summarization techniques and explain the methods used for each technique, quality of summary, and a few domains where summarizing has been applied. Furthermore, a theoretical background explanation on the main method on which the current study is based is offered. Additionally, this chapter presents the most significant evaluation metrics that are used in text summarization techniques.

2.2 Automatic text summarization

The automatic text summarizing system creates a summary, which is a shortened version of the content that contains a few important sentences. Text summarizing started in late fifties with Luhn's work [23] that automatically excerpts summaries from technical papers and magazine articles, and it has progressed significantly since then. In this field of research, wide varieties of methods and techniques have been developed [24]. Nonetheless, generating an automated summary is a difficult task. When summarizing a large number of documents, many issues such as redundancy, cohesion, cover the content, reflect the author's opinion, co-reference, and sentence ordering, and so on, require special attention, making the task more difficult [25].

The necessity for Automatic Text Summary (ATS) has arisen because of the increase in online publishing, enormous numbers of internet users, and the rapid expansion of electronic governance (e-government). Due to the rapid development of information communication technologies, a vast number of electronic documents are now available on the internet, and

users are having difficulty finding relevant information. Furthermore, the internet has made vast collections of text on a wide range of topics available. This explains why there is so much redundancy in the online texts. Users become so fatigued after reading a huge number of texts that they may overlook many important and relevant documents. As a result, in this phase, a reliable text summarizing system is required. These systems can condense information from a variety of publications into a concise, understandable summary[26][27]. Four main goals are considered by [28] : information coverage, information significance, information redundancy, and text cohesion.

2.2.1 Application of summarization

ATS is extensively used for the text mining techniques and applications of analytics like extraction and information retrieval, answering questions, and so on. Automatic text summarization systems is used in combination with information retrieval strategies to improve the abilities of search engines[29]. Every automatic text summarization system can accepts multiple text types. Thus, automatic text summarization systems are utilized for a variety of implementations such as email summarization, news summarization, legal summarization, biomedical document summarization, and so on. Below are some examples of ATS applications.

1) *News summarization*: is a textual summarizer, which it assists users in finding the most exciting news for them. Daily, the system will automatically collects news from various Internet news sites then clusters and categorizes them , finally summaries them.
2) *Sentiment summarization* : is the analysis of opinions of people, emotion, and judgment regarding products and events. It employs fuzzy logic that classify opinion or sentiments polarity in product

reviews such "Strong Positive," "Positive," "Negative," or "Strong Negative." It also uses the imputation of missing sentiment approach to integrate non-opinionated texts.

3) *Tweet/blog summarization*: Millions of messages are posted on social networking websites such as Facebook and Twitter. Microblogging websites like "Twitter" are crucial sources of real-time information during disaster events since hundreds, thousands, or even millions of microblogs (tweets) are posted on Twitter. As a result, microblog summarization has attracted increasing attention in recent years.
4) *Books summarization*: The majority of research concentrate on the summary of short documents. ATS systems designed for short documents are not appropriate for summarizing long documents such like books because: 1)the chosen sentences may well not cover all of the topics of book , 2)document length is necessary for good performance, and 3)using the feature of sentence's position differ in books from short documents.
5) *Email summarization*: Email messages are unstructured general text and not usually well coordinated. In[30] present an ATS system that employ linguistic techniques and machine learning techniques to identify noun phrases from Email messages and generate a summary.
6) *Legal document summarization*: An ATS system for legal documents is a system to an automatic search for legal documents that saves legal experts time. The summarizing task specifies various rhetorical roles involved in presenting the legal judgment document's sentences. Based on the legal query, the search task finds related previous cases.

7) *Scientific papers summarization*: Scientific papers are very well organized documents with several traits in common, such as the predictable positions of exemplary items in a document, basic words, and a structure like template. Automatic summarization is great importance here, for example, extracting a summary of the problem statement in a research paper and using it to find research papers related to this problem [31].

2.2.2 Automatic Text summarization system

Automatic Text Summarization systems could be divided into initial categories of single and multiple document summarization. In the single document summarization, only documents in similar theme and content is summarized efficient. While the multi-document summarization summarize documents from different fields and topics, where it is more complicate.

Furthermore, text summarization systems could be divided into Extractive and Abstractive. In the Extractive approach, some sub-sentences are selected from the original text sentences to represent the summarization of the original text. In other words, extractive selects some sentences from the original text without changing them. These selected sentence should give the reader full understanding of the original text. While in Abstractive summarization, the generated text is rephrased from the original text.

Figure (2.1) depicts the overall architecture for ATS, which includes three tasks

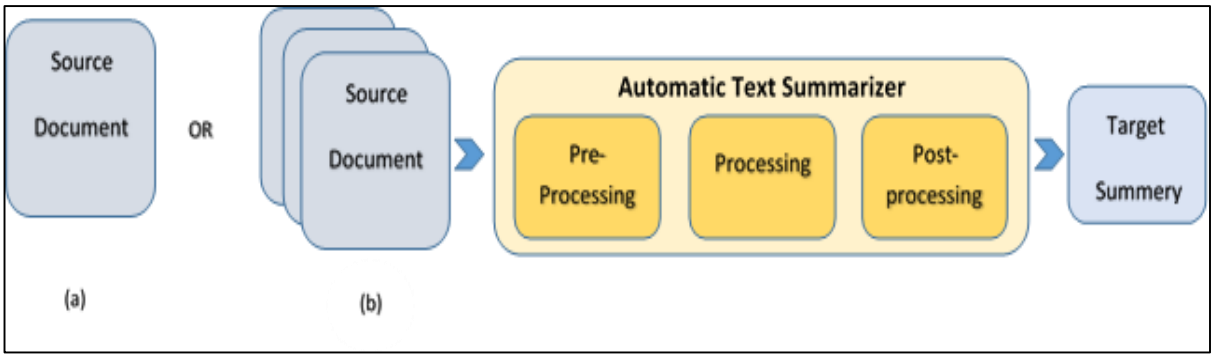


Figure (2.1) General architecture for ATS[109]

1. Preprocessing: in this phase, some operations are conducted to solve some problems and prepare the text for further process. The resulted text should be clean and structured properly for the next stage[32]. These operations include sentence segmentation, word tokenization, stop-word removal, part-of-speech tagging, stemming, and others.
2. Processing: utilizing one of approaches of text summarizing to generate a summary from the input document.
3. Post-processing: Before generating the ultimate summary, many issues in the sentences of generated summary such as semantics resolution and sentence reordering must be resolved.

The ATS is among the most difficult problems in Natural Language Processing (NLP) and, in general, Artificial Intelligence (AI). Some of the challenges presented by the text summarization systems are: 1) Select the most significant sentences in terms of meaning from the original text to represent it with a good understanding to the reader. 2) Large volume text such as books or newspapers introduce challenging task. 3) Multi-document summarization. 4) Evaluation of the summary that is generated by the computer without the requirement to compare it to the summary generated by human. And 5) Production of an abstractive summary more

similar to the summary produced by human[33].

Since about the beginning of Automatic Summarization Systems studies in the late 1950s, many attempts to generating summarized text by computers similar to human summarization.

2.2.3 Types of text summarization

There are several types of Automatic Summarization Systems (ATS) shown in Figure (2.2). These types are classified according to the criteria below:

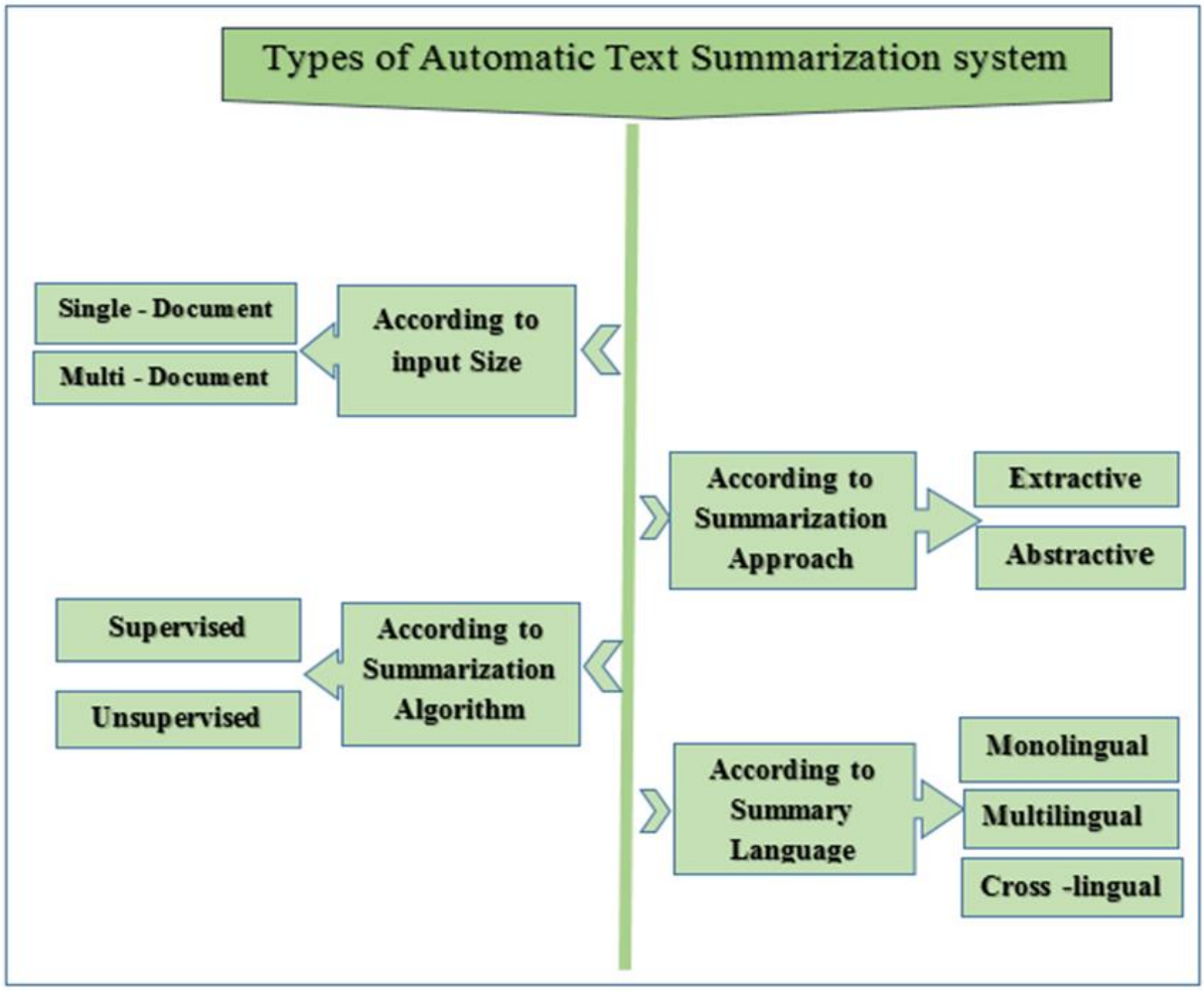


Figure (2.2) Type of the ATS[110]

A. According to the Input Size: Summarizations can be divided into three types based on the input text: single document, multi-

document and short text summarizations. The process of generating an abstract from multiple of documents is more difficult than summarizing a single document. The biggest issue that arises after summarizing multiple documents is redundancy, especially in the case of short texts. Some researchers dealt with the redundancy issue by selecting the first sentences in a paragraph and afterwards measuring their similarity to the subsequent sentences to choose the best one [34]. As a result, [35] recommends using the Maximal Marginal Relevance technique (MMR) to reduce redundancy. Several studies examine various methodologies and algorithms in order to obtain the optimum outcomes in multi-document and short text summarizing [21][36][37][38].

B. **According to approaches:** The summarization task could be classified as either abstractive or extractive. Depending on the length of the summary, an extractive summary is generated by selecting a few relevant sentences from the source text. Sentences in the original text are given some saliency features and scores, and the sentences with the highest score are chosen to form the final summary. On the other hand, an abstractive summary is the work of creating an abstract summary that incorporates new words that do not appear in the original text after the paraphrasing process. As a result, it is far more difficult than extractive summarization. [39].

C. **According to the Summarization Algorithm:** A summarization task, on the other hand, can be either unsupervised or supervised [40][41]. A supervised system requires the training data to identify the most influential content. Throughout the training phase, the algorithm requires a large amount of annotated training data for learning. These systems assign positive and negative labels to samples based on whether or not the sentence in the summary

appears[42][43]. For sentence classification tasks, common classification methods such as Decision Tree (DT) [44], neural networks [40], and Support Vector Machine (SVM) [39], are used. Unsupervised systems, on the other hand, do not need any training data. Just the original text is used to produce the summary output. several systems employ swarm and heuristic algorithms to generate highly relevant summary [45] [46][47][48].

D. According to summary Language: there are three types of summaries based on language: monolingual, multilingual and cross-lingual summaries [49]. A monolingual summarizing system is one in which the original source language and the summary document are the same. While a multilingual summarizing system is defined as one in which the source document contains many languages such as English, Arabic, and Spanish, and the final summary includes the same languages. But when the original document is written in Arabic and the extracted summary is written in English or another language other than Arabic, it is referred to as a cross- lingual summarizing system.

2.2.4 Text summarization techniques

There are two primary approaches to text summarization: extractive and abstractive. Figure (2.4) shows how each approach is implemented using various techniques. This section will give a comprehensive description of each of these approaches, as well as the methods used in the literature for each approach. There are several summarization methods available, including semantic-based, soft computing (SC)-based, graph-based, etc.

I. Extractive text summarization

To generate a summary, the Extractive based summarizing method chooses informative sentences first from document as they appear in the source document based on specific criteria. Figure (2.3) show the architecture of extractive summarization.

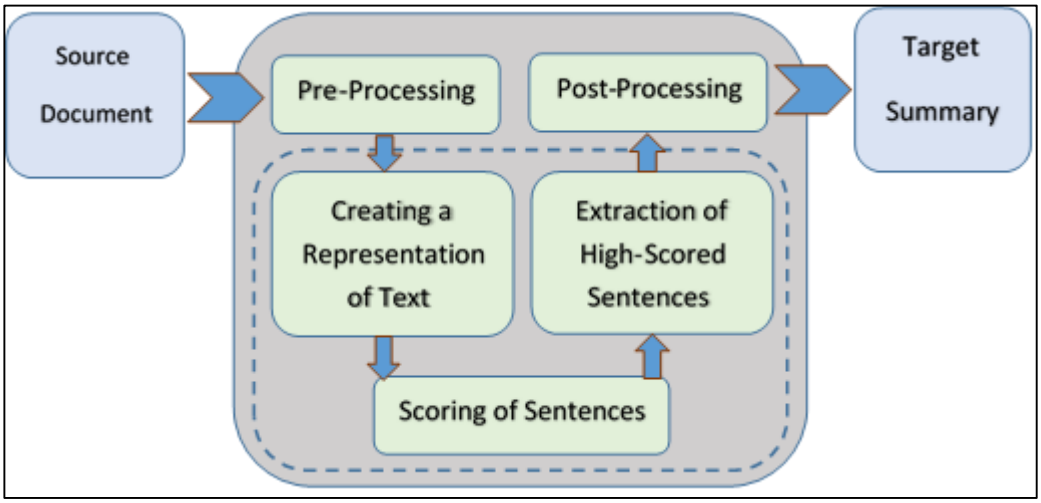


Figure (2.3) Architecture of an extractive text summarization system[2]

Before extractive summarizing, the main challenge is determining which sentences from the input document are significant and likely to be included in summary. Sentence scoring based on sentence features is used for this task [50]. First, a score is assigned to every sentence based on its features and then ranking the sentences based on their score. The highest-scoring sentences are most likely to be appeared in the final summary.

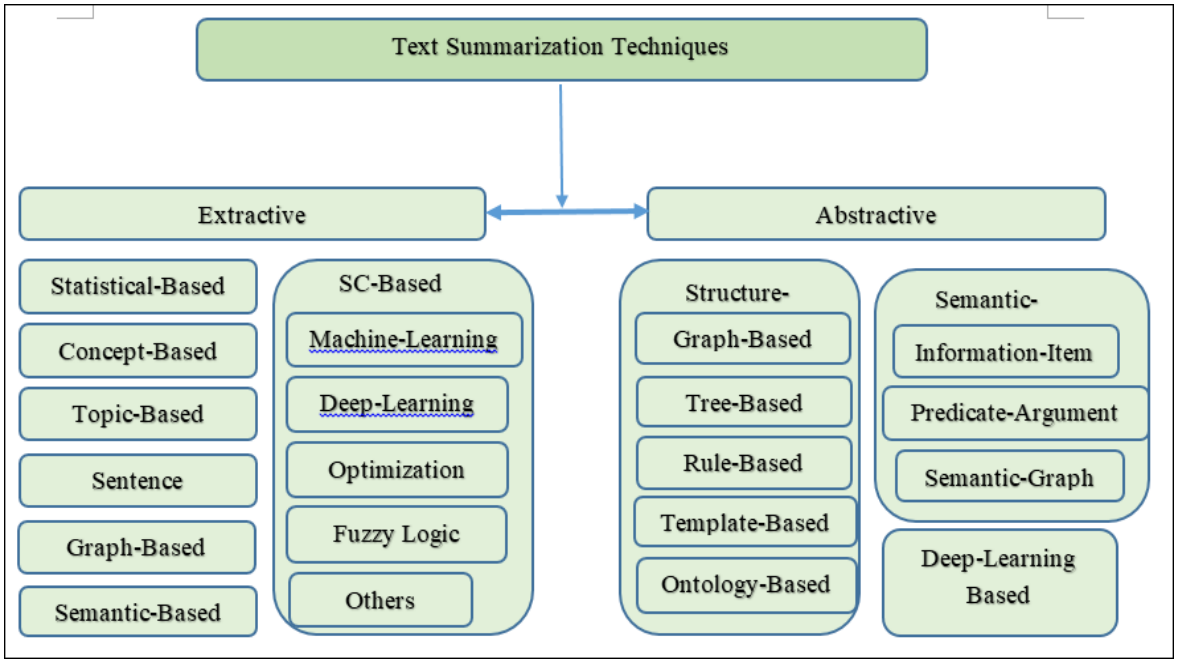


Figure (2.4) Automatic text summarization techniques and their associated

Figure (2.4) illustrates a variety of extractive text summarizing methods, such as statistical, concept-based, and so on. In this subsection, we will go through each of these methods and give a quick description of them.

A. Statistical-Based Methods: These methods use statistical analysis of a number of features to extract important words and sentences from the source text documents. To identify the "most important" sentence, some criteria's are known as the one that is "most favourably positioned," "most frequent," and other conditions [32]. The methods using statistical approaches are generally follows these general steps: 1) features extracted from the original texts are given different weights depending on the input text and how much they carry of these weights. 2) by using weight based formula, each sentence in the document is giving a score based these extracted features (all the selected features are included in the calculation in order to ensure a high quality summarization) [36].

B. **Concept-Based Methods:** These methods use external knowledge bases such as WordNet [51][52], HowNet, Wikipedia, and others to extract concepts from a piece of text. The significance of sentences is calculated through concepts acquired from the external base of knowledge HowNet Rather than words. To summarize the text in the alignment of the concept based summarization these following steps are followed: 1) retrieving text concepts from outside source of knowledge and data. 2) Constructing a graph model or conceptual vector to illustrate the relation between concept and sentences, and 3) scoring the sentences using a ranking algorithm [3].

C. **Topic-Based Methods:** the goal of these methods to identify the topic (or the subject) this document is covering. lexical chains, Term Frequency, Term Frequency-Inverse Document Frequency (TF-IDF) are popular methods which usually deployed to extract the topic of the documents. The steps used in determining the topic from the input document are: 1) converting the input text to an intermediate representation that reflects the input texts mentioned topics. 2) Assigning a grade to reflect the importance to every one of the sentences in the input document based on this representation [53].

D. **Clustering-Based Methods:** these methods consternate on dividing the sentences in the input document into different clusters. In which each cluster centre represent the most important sentence in the document. That this centric sentence will reveal a high level of information about the original text. The importance of each sentence is considered by the centre of its words. The usage of vector based method to calculate the centre of the document cluster is widely used. Usually TF-IDF is the most common method, with

a threshold to select the centres [54]. The following steps are used to calculate the scores for each sentence: 1) calculate TF-IDF for each sentence in the original text, to give a score of centrism for each sentence. 2) If the sentence hold more centric words than other sentences near them, it is considered a centric sentence [55]. The more the sentence close to the centre, the more important it is. As a text summarization approach, In the resulting summary, cluster-based summarization takes into account both relevance and redundancy elimination. The automatic summarization systems uses clustering techniques by following these steps of process. 1) cluster the original documents by using one of the clustering algorithms. 2) the sentences with important information will have higher score than others. Resulting higher rank for these sentences among the others. 3) the best sentences with higher rank and scores will be selected to represent the original text.

E. Graph-Based Methods: the techniques and methods used in this approach employ sentence-based graphs to express a document or a group of documents. For extractive summarization, such representations have been widely employed for example(LexRank[54]and TextRank [56]). Researchers Erkan and Radev describe how stochastic walks over sentence-based graphs may assist the process of summarizing documents and text [54]. According to Wang et al. [57] LexRank's sentence scoring phases are as follows: 1) using an undirected graph to represent the document sentences, with each node representing a sentence from the input document, and the weight of the connecting edge for each pair of sentences being the semantic similarities between the two related sentences applying the method of Cosine similarity. 2) determining the significance of each sentence by using a ranking

algorithm. The sentences are ranked according to their LexRank scores, much like the PageRan algorithm, except that the PageRank graph is directed [3].

F. **Semantic-Based Methods:** Latent Semantic Analysis (LSA) is a method using the technique of semantic summarization. Many automatic text summarization systems are using the approaches based on semantic similarity techniques. LSA is a Machine Learning approach of unsupervised learning, that represents text semantics according to the observed recurrence of words [53]. The sentence scoring phases for every LSA-based extractive summarizer involve [58]: 1) building the input matrix that include the term-to-sentence matrix, 2) identifying the relations between terms and sentences using Singular Value Decomposition (SVD) on the input matrix. Many other semantic-based techniques, such as Explicit Semantic Analysis (ESA) and, Semantic Role Labeling (SRL), are employed in ATS. Other researchers utilized the teqniques of SRL and ESA, besides the Wikipedia knowledge base to develop an automatic summarization system that is based on semantic extractive [59].

G. **Machine-Learning-Based Methods:** the approaches following Machine Learning transform the process of summarizing a text into a supervised classification problem. Considering each sentence as a record. By using the training dataset in Machine Learning model (which are the documents to train the model on), the model will learn the different patterns between the documents. And will be able to classify the sentences in the test documents effectively as a summary sentence or non summary sentence. This operation could be divided into two steps. 1) Find and extract patters from the features of the training dataset, which is the pre-processed

collection of documents. 2) pass these extracted features into Artificial Neural Network to discover the pattern and output the classification result [3].

H. **Deep-Learning-Based Methods**: some researchers offered a summarization method based on embedding's that employed document level similarity [60]. The meaning of embedding of a word is self explaining. Each document is considered as a bag. This bag holds smaller objects which are sentences. These sentences are considered as a bag of words inside each sentence. The similarity between the documents produce more meaningful summarization than those based on sentence level [61]. Researchers present an ATS system based on a learning approach and even a Recurrent Neural Network (RNN) sequence modeling of encoder-extractor network architecture for single-document summarization. The approaches based on the summarizing of sentence level is used and performed well in selecting the important features.

I. **Optimization-Based Methods:** the text summarization problem is reconfigured into optimization problem in these approaches. For example, in [62] multi objective optimization approach is used to develop an extractive text summarization system with multiple documents. The sentence scoring phases of optimization-based methods for extractive summarizer involve: 1) constructing a suitable representation for the input text, such as the widely used vector representation (wherein every sentence in the input text is expressed as a vector of words). 2) the sentences of the summarization are selected by using one of the optimization methods (for example Artificial Bee Colony with multiple objectives), based on the needed summary length limit, along with several optimization criteria such as: redundancy reduction, content

coverage, coherence and relevance. Moreover, Genetic Algorithms may be used in modifying the weights in the process of text summarization. The process of sentence scoring of system with extractive genetic-algorithm based, used for text summarization, according to [63], are as follows: 1) recognizing the features of the input text, such as sentence position, sentence length, and so on. 2) Adjusting the weights of these features using the genetic algorithm, then computing the sentence scores.

J. **Fuzzy-Logic-Based Methods:** in text summarization systems, Fuzzy logic is used as well. Since not everything in the universe could be represented as zeros and ones, the motivation behind fuzzy logic is to resembles human thinking and provides an efficient technique to express the feature values of sentences [64]. The steps for scoring a sentence are as follows [65]: 1) choosing a collection of features for each sentence, such as sentence length, term weight, and so on then adding the relevant rules into this system's knowledge base. 2) Using the fuzzy logic techniques to score the sentences based on the important of each sentences. As a result, a score value between zero and one is given for each sentence in the output, depending on the sentence features and the knowledge base's specified rules.

II. Abstractive text summarization

Abstractive summarization creates a broad summary by generating new phrases in the same way that a human being does. It is possible that the summary will include new phrases not included in the original text. Compression strategies and Language generation are required for generating abstractive summaries. The general framework of an abstractive text summarization steps are shown in Figure (2.5).

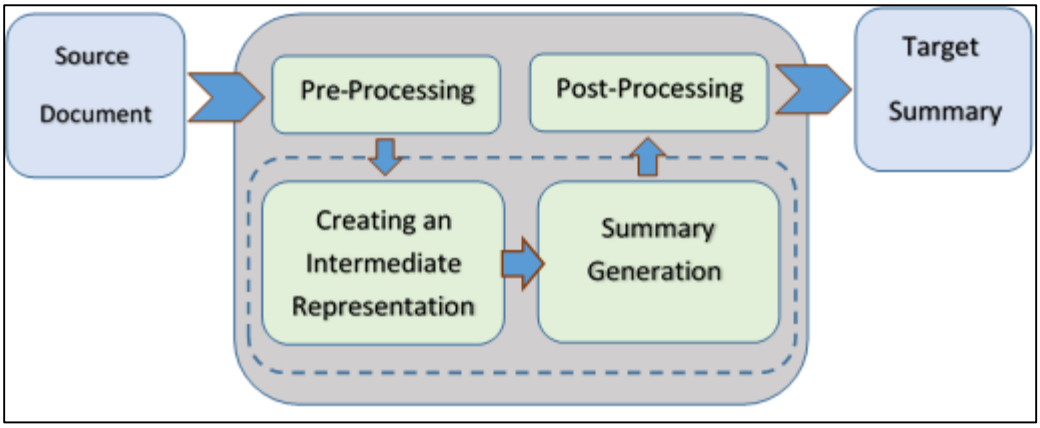


Figure (2.5) Architecture of an abstractive text summarization system [1]

Abstractive text summarization in general categorized into three types: Structure based approach, semantic based approach and deep learning approach. The structure-based uses pre-existed structures for example, trees, graphs, rules, ontologies and templates. In addition, semantic-based approach uses natural language generation techniques and text semantic representation for example based on predicate arguments ,information items, and semantic graphs While deep learning approach supply another classification to the abstractive approach as classical or neural-based which in general indicates to any system that is not based on neural [66]. The remainder of this session is devoted to listing and describing the mentioned techniques briefly.

A. **Graph-Based Methods:** Each node in the graph represents a word, and nodes are linked by positional information. Sentence structure is represented by directed edges. In [67]the graph-based method's processing steps involve: 1) creating a graph based on the textual nature, to describe the original text. 2) Producing the abstractive summary that you want. To do so, several sub-paths inside the graph are investigated and scored as below:

- Sort the scores of all the paths in descending order after ranking them.
- Using a similarity measure to remove paths that are repeated sash as Jaccard measure.
- The summary is created by selecting the top *k* remaining paths. And the *k* value indicate the number of sentences required for the summarization process.

B. Tree-Based Methods: in these approaches, the motivation is to find the sentences with similar information, and in order to provide an abstractive summarization, these sentences are combined together [68]. The used structure is based on Tree data structure to represent the similar sentences. Dependency trees, the most common tree-form representations for text, are built using parsers. Some tree processing operations, such as pruning, linearization (turning trees to strings), and so on, are undertaken to construct the final summary [66]. Some researchers [69]suggested abstractive summarizer for multi-document by following these steps: 1) transfer the corpus's in the input into texts to create a collection of the all the syntactic dependency trees. 2) Partially depended trees are extracted from the syntactic dependency trees. These extracted trees holds different sizes. 3) To ensure topical variety, cluster the collected partial dependency trees. 4) Single sentence is built by using the partial trees from each cluster. This sentence is the cluster inside the abstractive summary.

C. Rule-Based Methods: the approaches in this method necessitate the definition of rules and categories in order to identify the key concepts inside input text, which are then used to generate the summary. This method's steps are as follows: 1) Sort the input text into categories depending on the concepts and terms it contains. 2)

Generate questions depending on the input text's domain. 3) Inside the text, only relevant words and concepts will be used and selected to answer the questions. 4) The abstractive summary was created by feeding the answers into various patterns. For instance, the questions could be similar to "what is the event?", "who did the event?", "when the event has occurred?", "where the event has occurred?", "what was the impact of the event?" [66]. Some researchers suggested a solution in which they utilized the abstraction strategies to build the architecture for their solution in their paper [70]. Each abstraction scheme includes Information Extraction (IE) rules, content selection heuristics, and simple generation patterns to handle a certain subcategory or subject. The researchers carry out the creation of these rules manually. An abstraction scheme aims to respond to single or multiple aspects, and there may be multiple schemes that are connected to the same aspect. The IE rules can identify a number of candidates for each aspect, and the content selection unit can choose the best to transmit to the generating module.

D. Template-Based Methods: in real life, when the people summarize the documents, sometimes there is a general theme the people follow to summarize the texts. This theme is called a template. For example, if there is a fixed forum for some meeting or exam texts, some specific information will be accessed to give a summarization. The genre of the input document can be considered. Depending on it the summarization is created. And then using the data inside the input text to fill the empty slots in the correct pre-defined templates [4]. The text snippets used in the operation of filling the template are selected by using extraction rules and patterns of natural languages [68].

E. **Ontology-Based Methods:** almost all types of documents could be associated with predefined domains. Each domain contain some structure relevant to its specific nature. This structure of information could be transferred into knowledge dictionary, such as an ontology[71].The primary idea is to use an ontology to extract the necessary information from the source texts in order to create an abstractive summary [72].

F. **Semantic-Based Methods:** semantic representations is used to represent the input documents. These representations could be semantic graphs, predicate-argument structures, or information items. Which is then given to a natural language generation system, which uses verb and noun phrases to generate the final abstractive summary [66]. Some researchers [73] proposed an abstractive summarization system for multi-documents that have the following: 1) the input text is represented by using SRL with predicate argument structures. 2) Employs a semantic similarity measures to cluster semantically related predicate argument structures across the text. 3) Uses a Genetic Algorithm to rank the predicate-argument structures based on weighted and optimized features. 4) Generates sentences from predicate-argument structures using language generation.

G. **Deep-Learning-Based Methods:** Abstract summarization became possible after its recent success of learning from sequence to sequence (seq2seq) [74]. Seq2seq has performed well in a variety of NLP tasks, such as machine translation, dialogue systems ,and voice recognition[57]. A collection of RNN models showed a promising results for the summarization of short texts. These models are based on attention encoder decoder techniques. Yet, deep learning techniques still have some flaws, such as: 1) creating

repeated sentences or words. 2) incapacity to deal with words that are outside of one's lexicon. In [57], the summarizing system includes the following steps: 1) Transforming the dataset to plain texts, while keeping the documents in their original statues (e.g. news articles) as well as their summaries separately. 2) apply the segmentation of words and use the sub word model in dealing with the data. 3) the Gensim toolkit will be used to create the vectors of the words [75]. Which will be subsequently be trained in the proposed model. 4) Tensorflow will be utilized, with one bidirectional LSTM layer in the encoder side. And a unidirectional LSTM layer in the decoder side. The model will be trained by using Adam optimizer. Furthermore, cross-entropy will be used for the loss function.

To conclude, modern abstractive summarizing efforts have primarily focused on the use of deep learning models, particularly in the summarization of short texts[76]. Combining diverse methodologies and techniques for better abstractive summaries is recommended. Different text summarization algorithms generate different summaries from the same input texts, so combining outputs from several ATS algorithms to provide better summaries than summaries generated by individual algorithms [77]is quite promising. For hybrid summaries, the models based on the structure of the text are frequently combined with extractive based methods. While systems based on abstractive summaries, deep learning-based or semantic-based methods are usually utilized [66]. These methods, for example, could be part of the preprocessing stage. And it will select the key and important phrases as a preprocessing step. And then the abstractive summarization will be used to create the text summarization [66]. Some researchers present an abstractive based text summarization

system that includes data transformation step by using semantic approaches. And Deep Learning techniques that based on the encoder decoder method [76].

2.3 Vector Space Model

Vector Space Model (VSM) is a classical form of document representation that represents text documents as vectors. VSM is commonly used in Information Retrieval (IR) by representing documents and queries as vectors of weights. A document d consists of a set of terms (t_1,t_2…t_n) where each term is a word that exists in the document, and n is the total number of different words. Each term t has a corresponding weight w [78]. Such that each weight measures the significance of the term in the document or query respectively. The weights are computed based on the frequency of the terms in the query or the document such as the TF-IDF weights. The TF-IDF weight is the most common features[79]. It is stands for Term Frequency – Inverse Document Frequency, it has two parts to calculate. The motivation behind TF-IDF is to calculate how much relevant each word to its document[80].

The first part is the Term Frequency, which is simply calculates the frequency of each word in the document to the total number of words in that document. Equation (2.1) shows the equation for Term Frequency[79].

$$tf(t,d) = \frac{f_{t,d}}{\sum f_{T,d}} \qquad (2.1)$$

The second part is the Inverse Document Frequency. Which relates to the important of the term to its document. The motivation of the IDF is to give a little importance to common words (e.g. the, and, that). It gives higher importance to unique words, since they represent the core of the document more effectively. As shown in equation (2.3) [79].

$$idf(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|} \quad (2.3)$$

Where *D* is number of document divided by number of document *d* that contained the term *t*

Combining these two criteria's, the TF-IDF will calculate the importance of each term in its document. However, it will give less importance to the common words, and higher importance to unique terms. This will lead to relate the documents more effectively, based on the importance of the words. As shown in equation (2.4) [79].

$$TF - IDF(t, d, D) = tf(t, d) \cdot idf(t, D) \quad (2.4)$$

Where *t* is the term that exist in document *d* for multi document *D*.

Table (2.2) shows how to calculate the TF-IDF for each word in document in the example showed in Table (2.1). Next, the bag of words for each document is calculated.

Table (2.1) Example of number of document

Document ID	***Document***
D1	The best Italian restaurant enjoy the best pasta
D2	American restaurant enjoy the best hamburger
D3	Korean restaurant enjoy the best bibimbap
D4	The best the best American restaurant

Table (2.2) TF-IDF calculations example

Word	**TF**				**IDF**	**TF*IDF**			
	D1	**D2**	**D3**	**D4**		**D1**	**D2**	**D3**	**D4**
Italian	1/8	0/6	0/6	0/6	Log(4/1)=0.6	0.075	0	0	0
Restaurant	1/8	1/6	1/6	1/6	Log(4/4)=0	0	0	0	0
Enjoy	1/8	1/6	1/6	0/6	Log(4/3)=0.13	0.016	0.02	0.02	0
The	2/8	1/6	1/6	2/6	Log(4/4)=0	0	0	0	0
Best	2/8	1/6	1/6	2/6	Log(4/4)=0	0	0	0	0
Pasta	1/8	0/6	0/6	0/6	Log(4/1)=0.6	0.075	0	0	0
American	0/8	1/6	0/6	1/6	Log(4/2)=0.3	0	0.05	0	0.05
Hamburger	0/8	1/6	0/6	0/6	Log(4/1)=0.6	0	0.1	0	0
Korean	0/8	0/6	1/6	0/6	Log(4/1)=0.6	0	0	0.1	0
Bibimbap	0/8	0/6	1/6	0/6	Log(4/1)=0.6	0	0	0.1	0

The great advantage of VSM is the simplicity to search for documents or compare documents by using one of the similarity or distance measures like the Spearman distance, cosine measure, Euclidean distance, Vector inner product, Hamming distance, Correlation distance, or Jaccard distance. At retrieval time, the documents are ranked based on the cosine similarity values equation (2. 5) between the document vectors itself or between document vectors and the query vector and returned to the user from the highest to the lowest values.

$$\text{Cosim}\ (s_i, s_j) = \frac{\sum_{k=1}^{m} (w_{ik} \cdot w_{jk})}{\sqrt{\sum_{k=1}^{m} w^2_{ik} . \sum_{k=1}^{m} w^2_{jk}}}, i, j = 1,2.....,n \qquad (2.5)$$

Where cosim is cosine similarity function, W_{ik},W_{jk} are the TF-IDF weights of each word in sentences *i* and sentences *j*.

2.4 Sentiment analysis

Sentiment analysis is the task of automatically classifying texts according to the emotions they express. In the simplest scenario, we want to classify a text as positive, negative, or neutral. In more complex situations, we could identify specific emotions or compute the sentiment with respect to a specific entity.

There are lots of different ways to perform sentiment analysis, and using a dictionary is possibly the simplest one. A sentiment analysis dictionary contains information about the emotions or polarity expressed by words, phrases, or concepts. In practice, a dictionary usually provides one or more scores for each word. We can then use them to compute the overall sentiment of an input sentence based on individual words.

There are many of Sentiment Analysis Dictionaries such as SentiWordNet**,** SentiWords, and VADER**.** VADER is a lexicon and simple rule-based model to sentiment analysis. It can efficiently handle vocabulary, capital letters, abbreviations, symbols, recurring punctuation marks, etc. It is often used with social media platforms to express users' sentiment. This makes it suitable for analyzing sentiment-based texts on social media.

The result generated by VADER is a dictionary of four keys: *neg*, *neu*, *pos* and compound. *neg*, *neu*, and *pos* meaning negative, neutral, and positive respectively. Their sum should be equal to 1 or close to it with float operation. Compound corresponds to the sum of the valence score of each word in the lexicon and determines the degree of the sentiment rather than the actual value as opposed to the previous ones. Its value is between -1 (most extreme negative sentiment) and +1 (most extreme positive sentiment). Then the sentiment score of the word ($senti(wo_i)$) is calculated using sentiment lexicon. The context affects the polarity of the word in

which it appears. To address this, consider the following types of sentences:

- Objective and subjective sentences: The first focuses on facts and non-opinionated information based on data. While the second focuses on the author's opinion through the sentiment words it contains[81]. As a result, the sentiment score for objective sentences will be equal to zero.

- Interrogative and conditional sentences: These sentences do not represent opinions or express sentiments , even if they contain sentiment words [82]. Conditional sentences are distinguished by the conjunction "if," while interrogative sentences are distinguished by the character "?", which is discovered during preprocessing. Similarly, these sentences will get a sentiment score of zero.

- But-clause and negation sentences: The sentiment score for such sentences that include but-clause word or any negation may shift. These words, known as sentiment shifter words, have the ability to change a sentence's sentiment orientation[83]. Where if a sentence contains a negative word, the polarity of the next word is inverted. The listing of negation word has been acquired from[84]. The but-clause handling, on the other hand, is carried out as follows: if a but-clause word appears, in a sentence, the polarity, of all the words after it is inverted. That is, the polarity before the but-clause word and after it are diametrically opposed. The list of but-clause words has been provided by [85].

Finally, after assigning sentiment scores to words using the sentiment lexicon and determining the contextual polarity of the

sentence, the sentiment score of an opinionated sentence is calculated by the equation (2.6):

$$senti(s_i) = \frac{\sum_{j=1}^{sw_{si}} senti(wo_j)}{sw_{si}} \qquad (2.6)$$

Where *senti*(wo_j)*is sentiment score for word wo to index j in sentence* s_i , sw_{si} is the quantity of sentiment words in the sentence S_i.

The sentiment score of a sentence falls between [-1, 1]. Finally, the orientation of the sentence is described as following: If *senti* (s_i) > 0, the orientation is positive; if *senti* (s_i) <0, the orientation is negative; and if *senti* (s_i) = 0, the orientation is neutral. For more detail, see Figure (2.6)

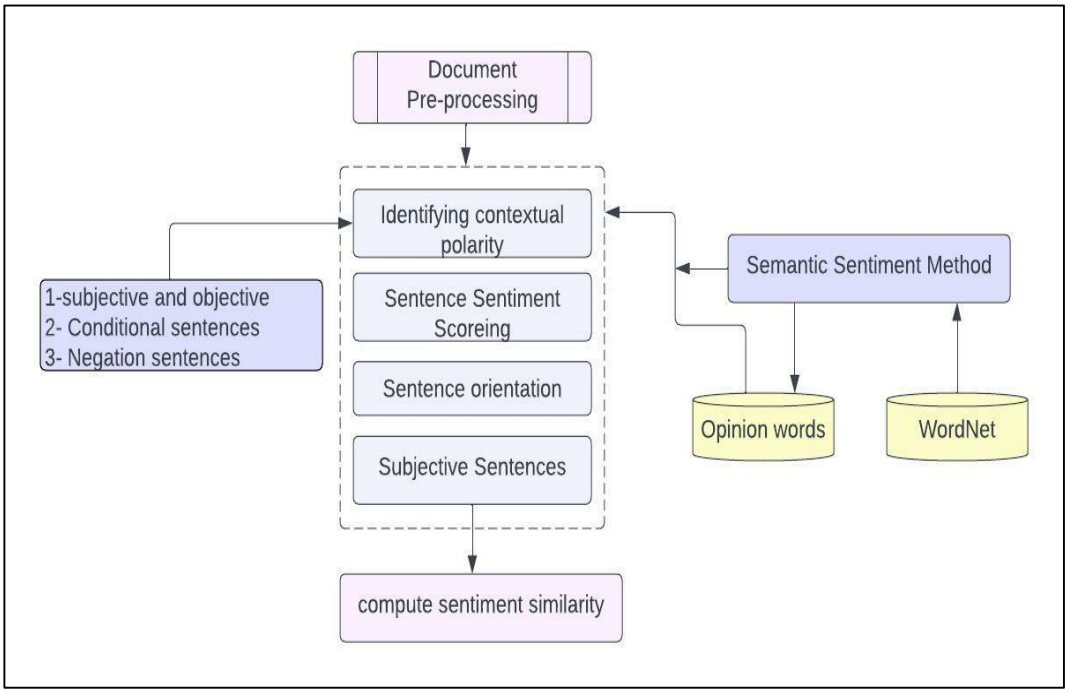


Figure (2.6) Sentiment analysis[18]

2.5 Objective functions in optimization-based summarization

There are some important objective functions that must be taken into consideration in the process of optimization-based summarization, such as:

A. Content coverage: The generated summary must contain the important sentences in the original document to cover its main content. This objective function must be maximized. This objective function is calculated according to Sanchez-Gomez et al. [11] as follows:

$$Content\ Coverage(X) = \sum_{i=1}^{n} cosin(s_i, O) \cdot x_i \qquad (2.7)$$

Where X is the document, $cosin$ is the cosine similarity function , S_i is the sentence in generated summary and O is the center of original document that is calculated as in equation:

$$O = \frac{1}{n} \sum_{i=1}^{n} W_i \qquad (2.8)$$

Where W is TF-IDF score for different word n.

While x_i is equal to:

$$xi = \begin{cases} 1 & where\ sentences\ is\ present\ in\ the\ summary\ generated \\ 0 & where\ sentence\ is\ obsend\ in\ the\ summary\ generated \end{cases}$$

B. Length of sentences: The generated summary should contain the shortest sentences in the original document, where the number of words in one sentence should be as small as possible. This function should be as minimal as possible. It is calculated according to Lucky and Girsang [15] as in equation:

$$L = word\ count\ (sentence) \qquad (2.9)$$

C. Redundancy reduction: The generated summary should not contain similar sentences. This function should be maximized as much as possible. According to Sanchez-Gomez et al.[10], the equation of the redundancy reduction is calculated as follows:

$$Redundancy\ reduction(X) = \frac{1}{\left(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} cosin(s_i,s_j).y_{ij}\right)\cdot\sum_{i=1}^{n} x_i} \qquad (2.\ 10)$$

Where *cosin* is the cosine similarity function between sentences S_i and S_j *in generated summary,*

While x_i and y_{ij} is equal to:

$$xi = \begin{cases} 1 & where\ sentences\ is\ present\ in\ the\ summary\ generated \\ 0 & where\ sentence\ is\ obsend\ in\ the\ summary\ generated \end{cases}$$

$$y_{ij} = \begin{cases} 1 & represent\ obscence\ of\ a\ pair\ of\ sentences\ Si\ and\ Sj \\ 0 & represent\ presence\ of a\ pair\ of\ sentences\ Si\ and\ Sj \end{cases}$$

D. Reflecting the author's opinion: The sentiment score of the produced summary should be close to the sentiment score of the original document, as it reflects the author's opinion of the original article. It should be maximized to the possible greatest extent. According to Sanchez-Gomez et al.[10], it is calculated as follows:

$$Sentiment\ Similarity = 2 - |sent(D) - sent(S)| \qquad (2.11)$$

Where *sent*(D), *sent*(S) are sentiment score of sentences for original document and summary generated respectively. These are calculated as in equations:

$$sent(D)\frac{\sum_{i=1}^{osD} sent(s_i)}{os_D} \qquad (2.\ 12)$$

$$sent(S) = \frac{\sum_{i=1}^{osS} sent(s_i)}{os_S} \qquad (2.13)$$

Where os_D , os_S are the number of sentiment scores of opinionated sentence in document *D* and in generated summary *S* respectively.

E. Content cohesion: The generated summary should contain coherent sentences by Calculate the cosine similarity between each sentence and all the other sentences[36]. It should be maximized to the greatest extent possible. The equation showed in (2.5).

Some optimization-based summarization problems employ a single objective, while others employ multiple objectives at the same time.

2.6 Ant Colony Optimization

Ant colony optimization (ACO) is inspired by some ant species' foraging behavior. These ants release pheromone on the ground to identify the most appropriate path for other colony members to follow. The ACO follows such a mechanism to solve optimization problems.

2.6.1 Combinatorial Optimization

Combinatorial Optimization (CO) problems include obtaining values for discrete variables in order to select the optimal solution for a given objective function. Many important practical and theoretical optimization problems are combinatorial in nature. For example, finding the shortest path between two cities in the Travel Salesman Problem (TSP), Choosing the best packet data routing on the Internet and Arranging jobs in the production line, whether in ports, airports, the automobile industry and others. The ACO can minimize or maximize, for example, the lowest cost or the highest production, as it depends on the objective function adopted in the problem[86]. Each case of the optimization problems can be represented in a triple set of space domain (Ω, F , S) where S represents a set of solutions to the problem (the optimal solution), while F is the objective function that was adopted for each of the solutions that were

found to the problem F(S) and the symbol Ω refers to the set of constraints which we take into account to solve this problem. whether it is the constraints of minimizing or maximizing according to the problem area, and the solution that achieves those constraints to the problem is called the optimal solution[87]. The approach taken to solve Combinatorial Optimization problems is to find a set of solutions and choose the best one based on the lowest cost, but with the grows exponentially of the number of possible solutions to the problem in proportion to the size of the instance n. In some cases, this approach becomes impractical with increasing computational efforts, even if the optimal solution is found.

For solving of combinatorial optimization problems, there are two types of algorithms: exact and approximation algorithms. Exact algorithms are algorithms that always find the optimal solution to a given optimization problem.

However, in combinatorial problems or total optimization problems, conventional methods are usually not effective enough, especially when the problem search area is large and complex. For most NP-hard problems, the performance of exact algorithms is not satisfactory. When the optimal results are not achievable, there should be a choice between optimality and efficiency. Since getting both of them is not practically possible. Meaning, getting good enough results in reasonable time, should be considered when the best solution will take very long time to find the optimal solution. Therefore, heuristic search algorithms could be the choice for this dilemma. Since the heuristic algorithms can get relatively good results with reasonable time.

The main premise behind the use of approximate approaches is to gain preferred results in terms of the optimal results, for complex problem cases at a fair cost. They are divided into specific heuristic and

metaheuristic methods based on their application. Specific heuristic methods, unlike metaheuristics, are developed to handle specific situations. Furthermore, because of the way they guide the search, they are not useful for real-world situations in practice. When the algorithm wastes run time on unpromising parts of the search space, they are more likely to fall into local optima. Metaheuristics, on the other hand, are general-purpose approaches. They are made to use high-level methods to get away from local optima.

Metaheuristics are the approaches and algorithms that are built to tackle different and large number of problems without requiring much adaptation. Both local search metaheuristics and the population-based metaheuristics are characterized based on whether they control a single solution or a group of solutions at each stage [88]. A single solution is manipulated during the search in local search-based metaheuristics, whereas an entire population is involved in population-based metaheuristics.

Metaheuristics have grown in prominence as a result of their successful application across a wide range of areas. The metaheuristics methods include Particle Swarm Optimization, Artificial Bee Colony and Ant Colony Optimization and others.

2.6.2 Behaviour of Real Ant Colony

Although some species of ants are blind, but through the chemical substance that they release, they can communicate with each other in the environment in which they live. Some types of ants release special substances called pheromones, such as ants belonging to the family Lasius Niger. In the natural world, some species of ants roam (initially) randomly, and upon finding food they return to their colonies after leaving trails in their paths called pheromone. If other ants find this path, they will

likely not continue to travel haphazardly, but instead follow the path, reinforcing it if they eventually find food. Over time, the pheromone thrown into the path begins to evaporate, reducing its attractive power. For that reasons, the short paths will have higher value of density. Evaporation of the pheromone offers an additional advantage, which is the prevention of falling in local optima. On the other hand, if the evaporation value is not used, the solution that is considered by the first ants will have higher probability, and tend to be used in the next generation. In which case space exploration to find better solutions will be constrained. The effect of pheromone evaporation on real ant systems is unclear, but it is very important in artificial systems. The overall result is that when one ant finds a good (i.e., short) path from the colony to a food source, other ants are more likely to follow that path, and positive feedback causes the rest of the ants to follow one path. This behavior is illustrated by[89], where the author conducted an experiment known as the double bridge experiments shown in Figure 2.7.a , 2.7.b. In the first experiment, where the ant colony is linked to a food source by two bridges of equal length at the beginning, the ants explored the surroundings of the colony and walked in random directions due to the lack of pheromone until finally reached the food source, as the ants deposit pheromone on the ground along the path they took between the food source and the colony. During the search, the ants choose one of the two bridges at random, and although the two bridges are of equal length, the concentration of pheromone on the two bridges will be different - albeit slightly - due to fluctuations Randomly, this will bring more ants to this bridge, which in turn will cause the accumulation of a greater amount of pheromone. In the second experiment, one of the bridges is twice as long as the other. At first, the ants also walked randomly on the tracks. The ants that choose the shortest path reached the source of the food faster, and therefore will

come back early to the nest, this leads to the accumulation of more pheromone on the shortest path and thus the colony converged to follow the same path. Then reinforced the experience by adding a new path that was shorter than the one that the whole colony followed[89]. The author found that the colony never discovered it and continued to follow the longer path and reinforced it more.

The conclude from this that the ants follow the optimal path, not the shortest, in their foraging behavior[90].

The idea of the ant colony algorithm is to imitate this behavior with an "ant simulator" that allows artificial ants to walk the graph and find the shortest solutions.

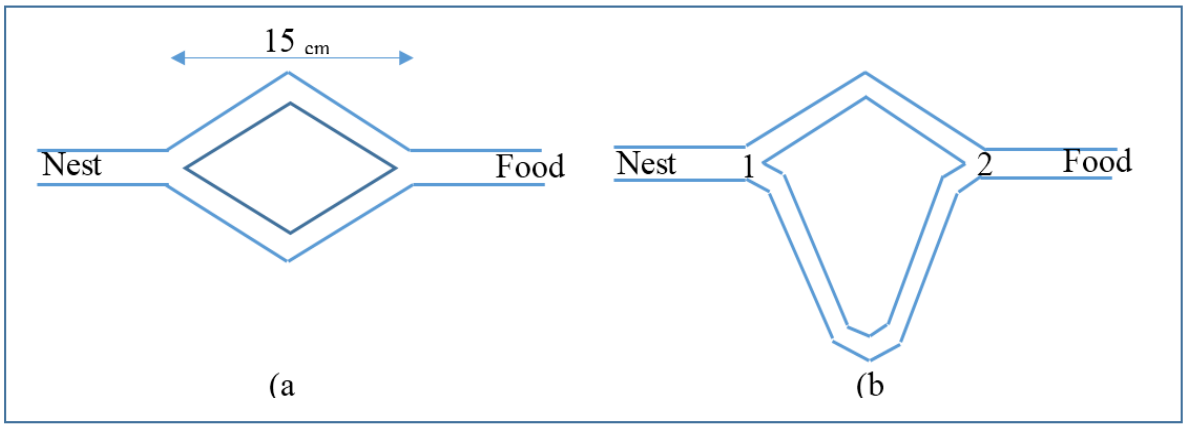


Figure (2.7) Double bridge experiment for Ant colony. a) Branches have equal length. b)Branches have different length[9]

2.6.3 Design of Artificial Ant

The probabilistic movement of ant colonies in the real world depends mainly on the intensity of the pheromone, they put it on their way to search for the food source and which they depend on finding the best (i.e. shortest) path between the original colony and the source of the food. The Ant Colony Optimization (ACO) algorithm uses agents called artificial ants that are similar in action to real ants in the problem of finding the shortest and optimal path in combinatorial problems.

There are some differences between real ants and artificial ants in terms of characteristics and behavior, which are explained in [91], which are as follows:

- According to the search for food, real ants evaluate the pheromone intensity in the paths they take from the nest to the food source directly, that is, the pheromone concentration in the shorter paths is greater than in the longer paths, while artificial ants evaluate the pheromone intensity based on some quality measures used in this field, which is used to evaluate the intensity of the pheromone in the path back to the nest.
- Real ants move in the environment asynchronously from the nest to the food source and may not follow the same route during return. While the artificial ants in each iteration of the system follow the same path in going to food and back to the nest.
- Real ants leave the pheromone on the way to and from the nest. While artificial ants do not deposit artificial pheromone on the way to the food source only on the way back to the nest.

The main objective of the Ant Colony Optimization (ACO) algorithm is to solve combinatorial problems. For this, some characteristics must be added to the artificial ant to increase its capabilities, which are not found in real ants[92] as follows:

- Save the paths that artificial ants follow during the process of building solutions is necessary to track the good paths, for this, a memory is added. The amount of pheromone reflects how good the given path is, the pheromone concentration increases when the solutions are good.
- The transmission of artificial ants through the paths they follow while searching for good solutions does not only take into account the amount of pheromone as in real ants, but also depends on the heuristic information.

- The pheromone evaporation parameter is added, this helps the algorithm to discover more solutions for the colony and prevents it from falling into sub-optimal solutions as a result of early convergence. While in colonies of real ants, this is not taken into account because evaporation is very slow and does not affect their search process.

2.6.4 Ant Colony Optimization Metaheuristic

Many optimization problems are difficult to solve by traditional methods, due to their large size, number of constraints, and complexity of the objective function. Such problems are classified (NP-hard) and cannot be solved in polynomial time. To solve these problems, it must be used algorithms that give approximate solutions to the optimal solution in a reasonable and acceptable computational time[93]. Therefore; metaheuristic algorithms are suitable for such complex problems. It builds the solution iteratively and cumulatively, as it exploits the entire solution space and diversifies the discovery of other areas of the solution space to get good and varied solutions. The ant colony optimization algorithm is one of those metaheuristic methods. It can be used to find approximate solutions to the problems of combinatorial optimization, and it is one of the methods of computational intelligence that is used by Dr. Marc Dorico for the first time in cooperation with Albert Collorni and Vittorio Manizzo. The idea of an ant colony algorithm for calculating the optimizations was derived from the method of access to the food source for real ant, as the ants are put a substance called pheromone on the ground in order to determine the best path that should be followed by the rest of the ants in the colony. In ACO, at first, the problem turns into a problem of finding the best path in a weighted graph with nodes and edges. The artificial ants build the solutions incrementally by navigating the graph from one node to another; it takes into account the heuristic information

and the amount of pheromone, as they are considered the main element in the success of ACO to solve optimization problems. The heuristic information is related to the diversity of the solutions, while the pheromone is related to the quality of the solution.

Figure (2.8) illustrates a conceptual representation for set of basic principles make it possible to increase the usage of Ant Colony Optimization approach to solve problems of Combinational Optimization (CO). First, a finite set C of solution components must be derived, it is used to construct solutions for the problem of the CO. Secondly, define set of pheromone values t, which knows as the pheromone model, this pheromone model is used to parameterize the probabilistic model. To construct the solutions in ACO problems, it should be combine the pheromone model $t_i \in t$ with component of solution.

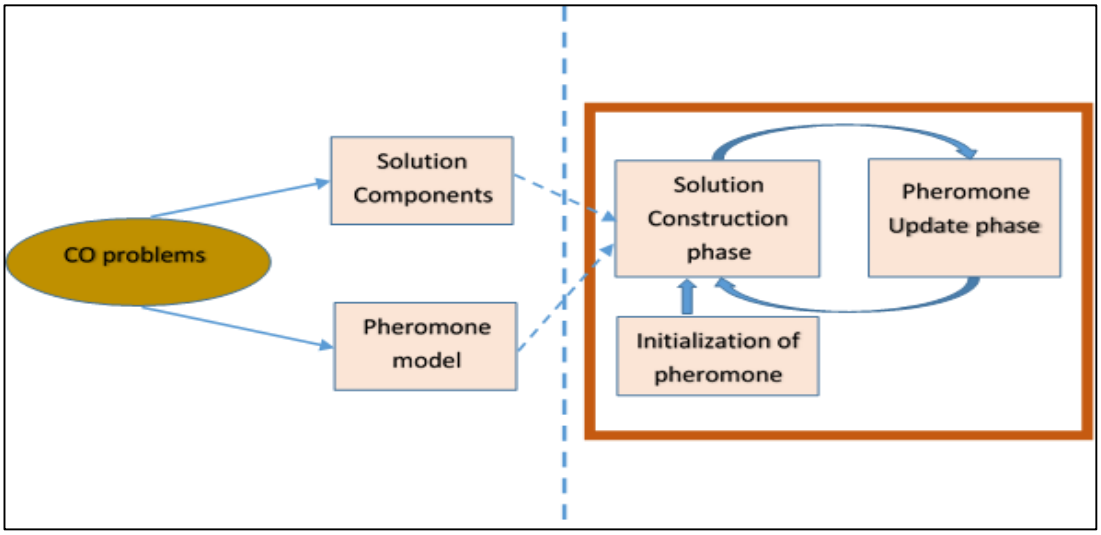


Figure (2.8) The conceptual framework of ACO[91]

The ACO method solves optimization problems iteratively in two steps:

- The pheromone model according to its probability distribution used to construct the candidate solution.

- The pheromone values' update by using candidate solutions helps to bias the sampling to get set of solution with high quality[91].

Informally, the ACO algorithm can be imagined as an interaction of three actions that appear in the pseudocode of the ACO metaheuristic in algorithm (2.1).ConstructAntsSolutions, UpdatePheromones, and DaemonActions [94]. Each action can be summarized as follows:

- ConstructAntsSolutions: Runs a colony of ants that randomly move across the problem graph. Artificial ants incrementally build solutions to the considered optimization problem and probabilistically, they take into account when moving to the neighboring node the heuristic information and the amount of pheromone. The ants evaluate the solutions after they finish building all their solutions, which is used to determine the amount of pheromone that will be added in the pheromone update phase.

- UpdatePheromones: During this stage, the pheromone trails are updated. The update may be an increase or decrease in the intensity of the pheromone in the trails. Pheromone's Decreasing can be accomplished out of pheromone evaporation. Evaporating of the pheromone helps exploration and avoids stagnation. The pheromone is increased by depositing a new pheromone amount on the paths used by ants while building solutions. The paths that are employed in better solutions receive more pheromone. Where the intensity of the pheromone is higher at the best solution that is found.

- *DaemonActions*: is an optional phase in which the original solutions are enhanced or maybe a centralized action that cannot be performed by a single ant is implemented. For example, Activate local optimization on solutions or Extra pheromone may be applied to the best solution obtained so far.

Algorithm (2.1) Algorithm of ACO metaheuristic[95]

Algorithm Ant Colony Optimization
Set parameters ,initialize pheromone trials **While** (termination condition not met) **do** Solution Construction Pheromone Update Daemon Actions // optional **End while** **return** best solution

2.6.5 ACO Problems representation

ACO algorithms are excellent choice for addressing combinatorial problems because the artificial ants construct the solution by adding one component at a time. Because ACO methodologies may be run continually and modified to changes in real time, they are also ideal for problems where its environment may change dynamically.

According to[96][91] the following features must be defined for a problem to be classified as an ACO problem:

- Exist a finite set of components for the problem C={c1,c2,…,cn}.
- Determine a set of constraints Ω of the problem to be solved.
- The states of the problem can be expressed as a finite sequence of components q=<qi,qj,…qu>. Let Q denote the set of all sequences q and O denote the set of feasible solution in Q satisfying the constraint Ω.
- Exist set of candidate solution S where S ⊆ q.
- Exist set of feasible solution S* where S*⊆q.
- Exist a non-empty set of optimal solutions N where N ⊆q.
- Exist an objective function (s) to evaluate the cost for each

solutions in S.

Therefore, the search space in the ACO can be represented as graph $G = (V, E)$, in which V is the nodes to visit, and E are the edges established between these nodes. [97].

The goal of each ant is to find the shortest path, starting from node v and search in G. These steps are stored in order to compare and find the shortest path. Each ant will look for the solutions in the near by nodes. This search will be based on the pheromone value and any other criteria is added according to the problem. After visiting all nodes, the solution is evaluated by following the objective function, and build a solution.

2.6.6 Ant Colony Optimization Components

This algorithm is developed as a consequence of various experiments on the foraging behavior of real ants. The experiments are carried out by [90].

The goal of the travel salesman problem (TSP) task is to identify the best (in terms of minimum length) path between cities. The problem state that the agent who travels these cities should start and end at the same predefined city. Another restriction is that each city should be visited only once. This task is extremely readily suited to the concept of the Ant System because their concepts are so similar: determine the shortest route between two nodes in a graph.

The ACO system covers a single ant colony composed of m artificial ants that collaborate with one another. Each arc connecting two distinct cities is assigned a specific amount of pheromone t_0 before the algorithm begins to run. This value is usually low. Its main purpose to make the edge selected greater than zero.

The algorithm is divided into two phases: tour/solution construction then pheromone update. Other critical decisions must be taken before the

ants may begin searching for a solution, such as defining the solution's structure (representation) or the initial pheromone amount to be assigned to each arc[98].

A. Initialization:

The initialization of the values of the arcs between the nodes are set based on two values. The first one is a variable value that represents the pheromone. The pheromone is given relatively a small value, and similar to all solutions. This value will be updated later depending on the best solutions discovered by the ants.

The second value is the Heuristic information, which will be a static value between the nodes. It is the inverse value of the distance between each two nodes. It is calculated by the equation (2.14) [99].

$$\eta_{ij} = \frac{1}{d_{ij}} \qquad (2.14)$$

Where η_{ij} is the heuristic information or visibility value between two nodes i,j. In TSP problem, the heuristic information equal to inverse of distance between two cites (i,j). As shown in equation (2.10).

As a result, the cities nearest to the ant will have a bigger visibility value, while the others will have a smaller one.

B. Solution construction: At each iteration, each ant is randomly assigned to one of the n cities. That city will serve as the beginning point for the tour that the ant will construct. A TSP solution can be represented by a series of n successive cities. As a result, the ant must choose the next city to visit with a certain probability at each stage of the construction.

Notice that the problem of Text Summarization of this work will be reformulated into TSP. in which, each sentence in the document is represented as a city. The connecting relationship between each sentence is represented in the arc between cities. In the following, the term city will be used to point into the sentence.

Each value for each arc is constructed for each ant by using a probability equation. The ant will choose the next city by employing a transition rule, which is short for random proportional transition rule, which contain a mixture of the attractiveness of the city, which is provided by the problem's heuristic information η_{ij}, and the fitness of the move, which is provided by the pheromone amount T_{ij}. The transition rule calculates the probability of ant k, located in city i moving to city j and is provided by the following probabilistic rule in equation (2.15) [100].

$$P_{ij}^{k}(t) = \frac{[T_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{\iota \in J_i^k} [T_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}} \qquad (2.15)$$

Where J_i^k is the set of cities that ant k has not yet visited while in city i , α and β are weighting parameters indicating the relative relevance of the pheromone and the heuristic information, respectively.

The values α and β are two adjustable parameters that weight pheromone and heuristic information in that transition rule.

C. Update pheromone: The values of pheromone in the arcs are updated when the solutions are constructed. The update occurs in two stages. The method applies an evaporation rate p to the pheromone present at each arc just before the ants can deposit pheromone in the arcs of their solution, with $p \in [0, 1]$, as in equation (2.16) [95][102] .

$$T_{ij}(t)_{new} = (1 - p).T_{ij}(t)_{old}. \qquad (2.16)$$

This action simulates the natural evaporation process, it avoid the algorithm from converging too soon (all ants building the very same tour) and being trapped in a local optimum. The evaporation rate *p* value reflects the relative relevance of the pheromone values from one iteration to the next. If *p* close to 1, the pheromone trail will have no permanent influence, allowing for more exploration of the solution space. While a small value will raise the importance of the pheromone, allowing for more exploitation of the search space near the current solution.

The length $S^{K(t)}$ of every tour is then computed, and the ants are permitted to deposit pheromone in each arc of their tour. As shown in equations (2.17) and (2.18) [101], the amount of pheromone deposited per each arc is related to the quality of each ant's solution and the number of ants that integrate that arc in their solution.

$$\Delta T_{ij}(t) = \sum_{k=1}^{m} \Delta T_{ij}^{k}(t) \qquad (2.17)$$

$$\Delta T_{ij}^{k}(t) \begin{cases} \frac{Q}{S^{k(t)}} \; if(i,j) \; belongs \; to \; the \; solution \; of \; ant \; K \end{cases} \qquad (2.18)$$

Where *Q* is a positive parameter and $S^{K(t)}$ represents the length of the tour built by ant *k* at iteration *t.* This update reduces the search space for tiny problem instances, bringing it closer to one in which optimal solution components get the highest values in the matrix. However, for big instance problems, stagnation is likely to occur, leading to a suboptimal solution instead of an optimal one. That is why pheromone evaporation is extremely important.

D. Stopping criteria: The preceding steps are repeated whenever some stopping criteria is met, which can be a defined number of

iterations, but it can also be the setting of a running time bound or maybe the number of solutions assessed.

The best values for the parameters utilized in ant algorithms are determined by both the nature of the issue and the approach used to search the solution space. As a result, decisions on the search technique must be made before selecting values for the parameter. The algorithm must then be executed numerous times to determine the values of the parameters that head for perform better.

2.7 Evaluation Metrics

The evaluation of a summary is a complex procedure because there is no idealistic summary for a document or set of documents, and indeed the definition of a good summary is mostly unclear. Furthermore, the wide use of different metrics, and the absence of a standard evaluation measure, has made summary evaluation difficult and complex.

Four main goals that should be taken into account in order to create readable and a concise summary: 1) Information coverage, 2) Information redundancy, 3) Information significance, 4) Coherence.

Evaluation of text summarization can be done manually or automatically[103].

2.7.1 Manual Evaluation

Evaluation of computer-generated summaries by human judges is based on several criteria[103]:

- Readability: Check the summary's linguistic quality to ensure that there are no missing data in its language structure.
- Coherence: The summary should be comprehensive and related in its content. Therefore, the sentences have to be structured and organized in the right way.
- Content coverage: The summary have to cover in general the

aimed content of the original document.

- Non-redundancy: Summary generated should not involve redundancy and repeated sentences. Which might take the shape of complete words or portions of sentences being repeated.
- Conciseness: the generated sentences should be related in its information to the former and later sentences.

Summarization analysis and Manual evaluation takes effort and a long time because it requires humans must read both the summaries and the original documents.

2.7.2 Automatic Evaluation

This subsection looks at the most often used evaluation measures in the literature.

A. Rouge family

One of the most used and effective metrics is ROUGE. ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. Which in general, is a set of methods used for comparing how good a text summarization or translation produced by machine with the text produced by humans.

There are few different types of Rouge metrics that each one follow slightly different approach than the other. In the following, each one will be discussed[104].

There are few different types of Rouge metrics that each one follow slightly different approach than the other. In the following, each one will be discussed.

i. Rouge-N

The Rouge-N calculates the number of matching words or tokens between the machine generated and human generated (i.e. reference) text. The N in Rouge- N refers to 'n-grams'. If the number of n-grams is 1, it is referred to as unigram. In which each single word in the reference text is searched for in the generated text.

If the n is 2, this is referred to as bigram. In which every two words are matched together between the original and the generated text. Similarly, if the n is 3, it is referred to as trigram [104]. Figure (2.9) shows an example of the Rouge-n metric.

To measure the Rouge-n, if we use unigram (i.e. Rouge-1), we would calculate the matching rate between the generated text and the reference text.

Figure (2.9) An example of the different Rouge-n metric

ii. Recall

The recall counts the overlapping number of n-grams in the generated text and the human-generated (reference) text, divided by the total number of n-grams. The equation (2.19) shows how to calculate it.

$$recall = \frac{count_{match}(gram_n)}{count(gram_n)_{refrence}} \qquad (2.19)$$

The problem of this metric, is that it will give a high value if the generated text is covering the whole words in the reference text. Even if there are many irrelevant words in the generated text [104]. Figure (2.10) illustrate this problem.

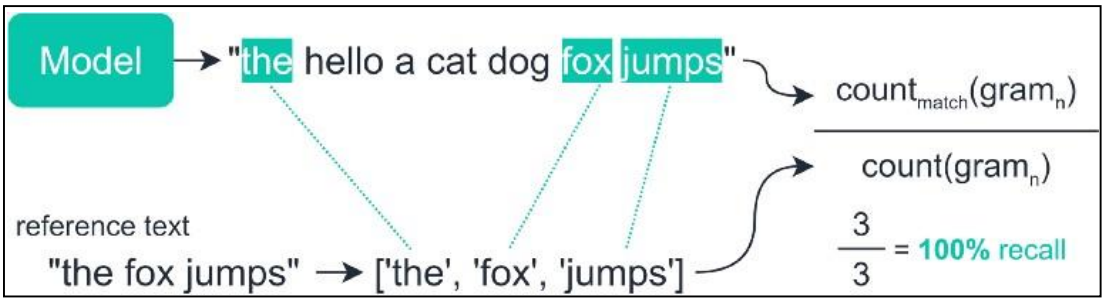


Figure (2.10) Example of the limitation of Recall metric

iii. Precision

To overcome the problem of the recall metric, in precision similar equation is used. But instead of dividing by the total n-grams in the reference, it is dividedby the total n-grams in the generated text. As shown in equation.

$$precision = \frac{count_{match}(gram_n)}{count(gram_n)_{generated\ text}} \qquad (2.20)$$

Figure (2.11) shows the same example used in the recall metric, but it is calculated by using the precision metric [104].

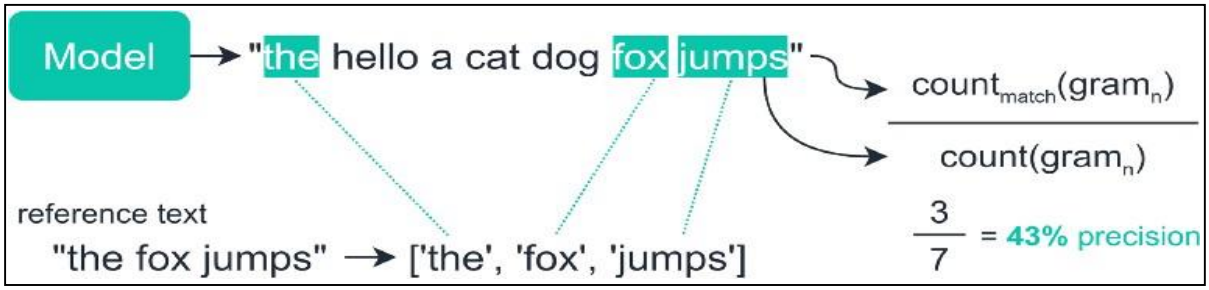


Figure (2.11) An example of precision metric

iv. F1 Score

The F1-score is used to combine both the recall and precision metrics. Equation (2.21) shows how to calculate it.

$$F1 - score = 2 \times \frac{precision \times recall}{precision + recall} \qquad (2.21)$$

Figure (2.12) shows how the used example previously will calculate the F1-score for it.

The F1-score is calculating the coverage of the original text (by using recall),and making sure it is not covering irrelevant words (by using precision).

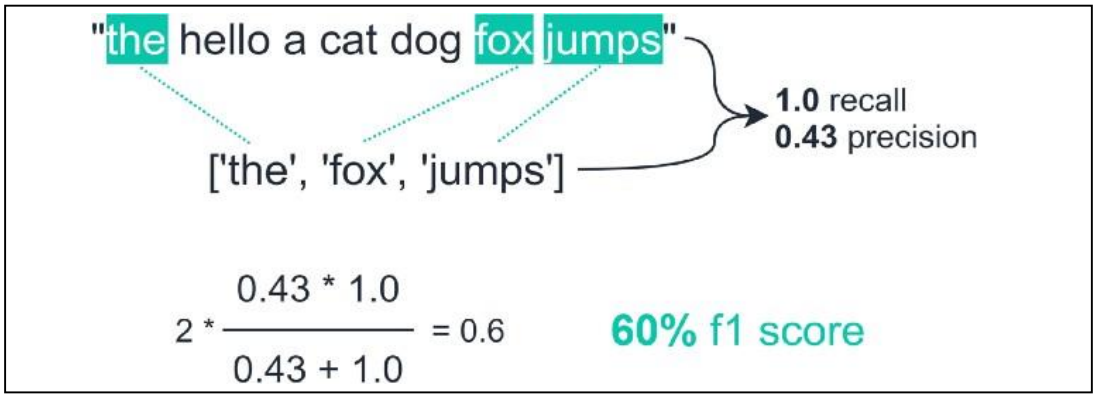


Figure (2.12) F1-score example

v. Rouge L

In Rouge-L, the metric uses the Longest Common Subsequence (LCS) to measure the recall, precision, and F1-score instead of the n-grams. The LCS is the longest attached words that appears together in the generated and reference text. Figure (2.13) shows how to calculate it [104].

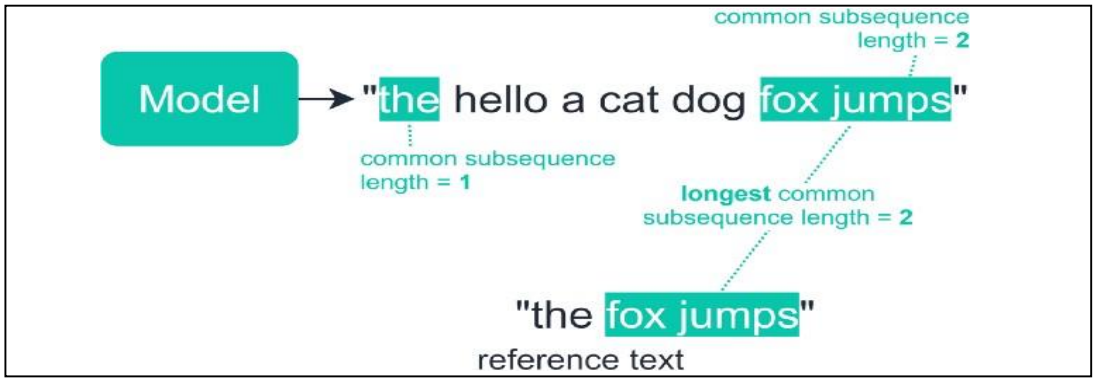


Figure (2.13) Example of LCS calculation

The motivation behind the LCS is the longer the shared sequence is, the more similar the texts are.

Figures (2.14) and (2.15) shows how to calculate the recall and precision respectively based on the given example.

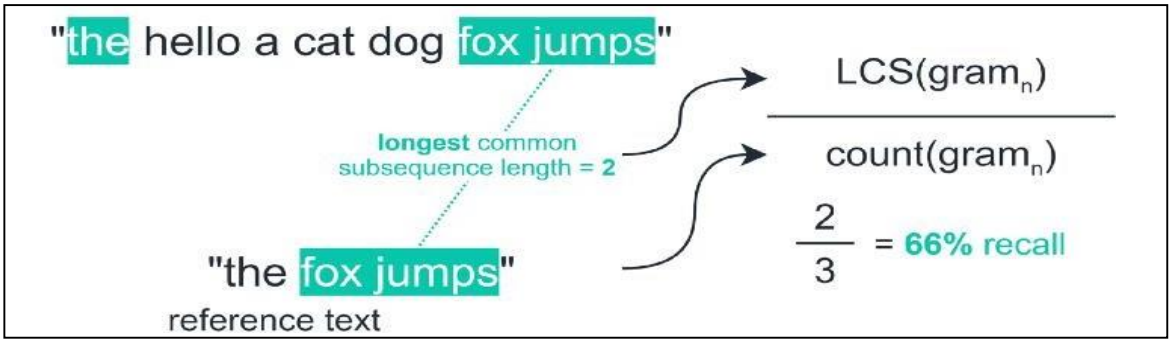


Figure (2.14) Recall calculated based on LCS

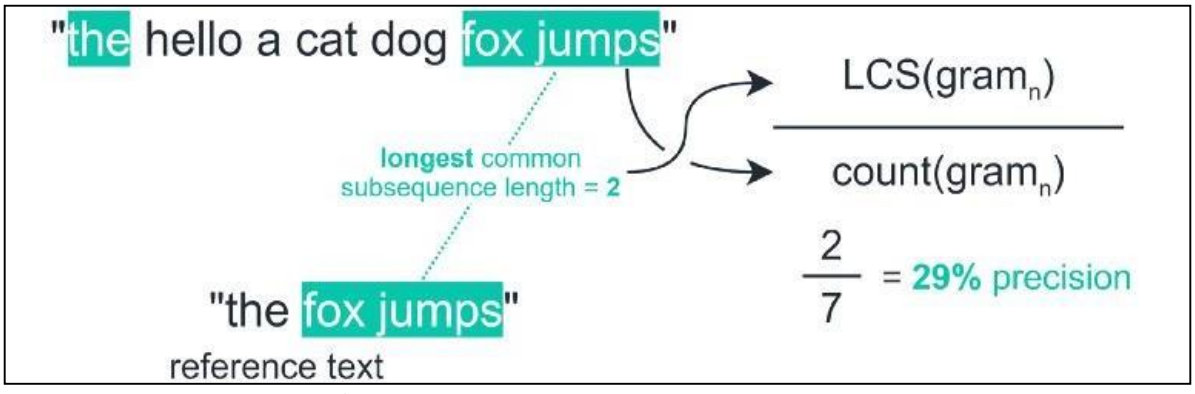


Figure (2.15) Precision calculated based on LSC

Figure (2.16) shows how to calculate the F1-score based on the LCS recall and precision.

Figure (2.16) F1 score calculated based on LCS

B. Statistics measure

Statistical measures are a set of statistical operations. Their main use is to analysis all data in order to help in obtaining meaningful results. Statistical measures have an important and effective role in scientific research as they are used as basic rules to ensure that the results are consistent with the main objective. A set of arithmetic operations are performed through the scale tools and data collection, and then the steps of the arithmetic operations are started[105].

The statistical measures that are used in statistical analysis can be classified as follows:

i. Measures of central tendency

Measures of central tendency are used to describe the gathering point of all data[105]. The most important of which are:

1. mean

It is one of the statistical measures that can be calculated in a simple way by calculating the summation of the total data divided by the number of data for the same variable.

One of its advantages is that it takes into account all values. One of its disadvantages is that outlier values can affect the calculation equation, and it is used to calculate descriptive data only.

2. Median

The median is defined as the value that is found in the middle of the values after they are arranged in ascending or descending order. Where all values are divided into two equal parts. One of its advantages is that

outliers do not affect it. Among its disadvantages is that it does not consider all values.

3. Mode

The mode is defined as the most frequently occurring value of all the data. One of its disadvantages is that it does not take into account all the values of the data set.

ii. Measures of dispersion

Dispersion measures are frequently used to analysis data to obtain insight about how sparse the data are, and how far they are separated. The most important of which are:

1. Range

It is the distance between the maximum and the minimum value in the variable in the data set, where there is no information about the mean of the data. One of its advantages is that it is easy to understand and calculate, but it is not considered a reliable measure for calculating the dispersion and it depends entirely on changing the scale.

2. Standard deviation

It is a good measure of dispersion as it provides information about the mean and the deviation from it. Its value becomes greater when the distribution of marks becomes heterogeneous. Equation (2.22) shows the calculation method for the standard deviation[98].

$$\sigma = \sqrt{\frac{\sum(x_i - \mu)^2}{N}} \qquad (2.22)$$

Where: σ is the standard deviation of the data. N is the population size, x_i is each value in the population, and μ is the mean of the population.

iii. Correlation metrics

It is one of the most important statistical measures. It highlights the strength of the relationship between the variables with each other. One of the most important of these measures is the correlation coefficient, which is the real measure of the nature of the linear relationship between two variables. The value of the correlation ranges between [-1,1].The closer the value of the correlation coefficient is to -1 or 1, the stronger the correlation, where it tends to be on a straight line in the scatterplot model. There are many types of these metrics, the most important type is the Pearson correlation coefficient. It is calculated according to the following equation[98]:

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}} \qquad (2.23)$$

Where: r is the correlation coefficient, x_i are the values of the x-variable in a sample, $\bar{x}$ is the mean of the values of the x-variable, y_i is the values of the y-variable in a sample, and $\bar{y}$ is the mean of the values of the y-variable.

Chapter Three

The Proposed System

The Proposed System

3.1 Introduction

This chapter describes the methodology and system design used for solving the described problem in chapter one. It presents the implementation of two factors. First, sentiment similarity by using Vader lexicon. Second, the TF-IDF similarity. These two factors of the sentences are used as the distance similarity in the Unified-objective ant colony optimization algorithm to achieve the research objectives.

3.2 Architecture of the Proposed Methodology

The architecture of the proposed methodology involves five main stages. These stages are data reading and preprocessing, similarity analysis, sentiment analysis, Ant Colony Optimization, and evaluation. As shown in Figure (3.1).

The first stage of data reading and preprocessing, starts by reading the DUC 2002 dataset. Furthermore, it includes sentence segmentation, word tokenization, Stop words removal, and Stemming.

The second stage of similarity analysis, involves calculate text similarity based on vector-based word and TF-IDF (Term Frequency–Inverse Document Frequency) similarity measure, and generate construction graph.

The third stage of sentiment analysis, involves calculate sentiment score for all sentences in the document, and identify the orientation of the sentences. This operation is carried out by using the Vader Lexicon.

The fourth stage includes this problem using Unified Objective Ant Colony Optimization.

And finally the last stage, include evaluate the work by using Rouge 1, Rouge 2, and Rouge-L. Furthermore, the cosine similarity, redundancy reduction, and sentiment similarity are calculated between the summary that generated by humans in DUC2002 dataset and the summary that generated by SO-ACO in order to evaluate the text summarization.

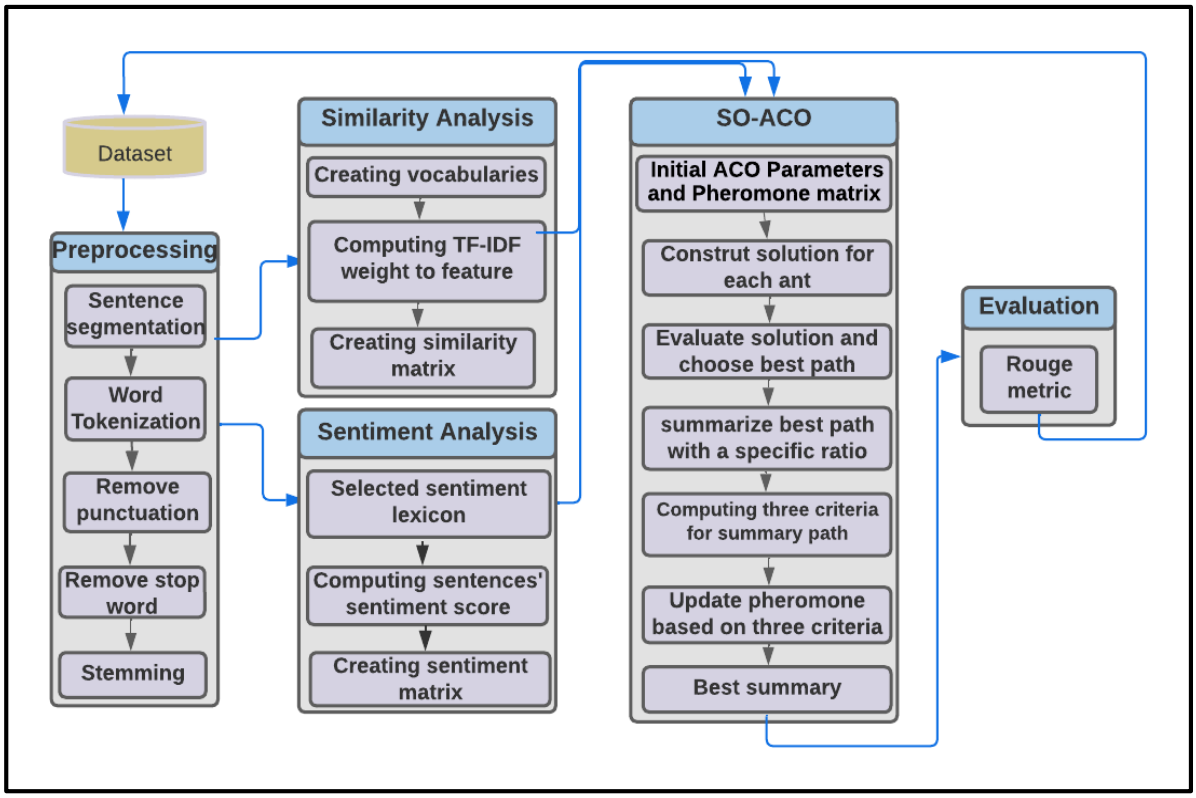


Figure (3.1) General overview of the proposed system

3.3 The Proposed algorithm

This work proposes a summarization algorithm that generates extractive summaries of a set of documents. This algorithm begins with a pre-processing of the input data. Next, it calculates the score of similarity and sentiment for each sentence in the document. These scores are used to create summaries that cover three main criteria: coverage of content, reduction of redundancy, and reflecting the author's opinion using the (ACO) algorithm. The proposed algorithm for this work (denoted as SO-ACO) is explained in the Algorithm (3.1)

Algorithm(3.1) SO-ACO pseudocode
Input: Document D
Output: set of best solution S
// Preprocessing steps// *1: sentence segmentation (D)* *2: Word Tokenization (D)* *3: Stop words removal (D)* *4: stemming (D)* *//Text similarity analysis steps//* *5: Weights ⟵ Calculate Tf-idf Matrix (D)* *6: O ⟵ Calculate Mean Vector(Weight,D)* *7: CoSims ⟵ Calculate CoSims Matrix(Weight,D)* *//Sentiment analysis steps//* *8:WSentimS ⟵ calculate words sentiment scores(D)* *9:SSentisS ⟵ Calculate Sentences Sentiment Scores (WSentimS,D)* *10:DSentiS ⟵ Calculate Document Sentiment Score (SSentisS,D)* *//Unified-objective optimization steps//* *S ⟵ 0* *11: initialize parameters for ACO algorithm* *12: While termination condition is not met* *13: For each ant in all ants* *14: select next sentence by construction solution equation (2.15)* *15: End for each* *16: Evaporate Pheromone by evaporate equation(2.16)* *17: Get ants which get best solutions* *18: For each ant in best solution* *19: Deposit pheromone by deposit equation(2.17)* *20: End for each* *21: S ⟵ filter solution*

3.3.1 Pre-processing of dataset

The input document set *D* can be represented first as group of *N* documents, $D=\{d_1,d_2,d_3,\ldots,d_N\}$, and each d_n in *D* can be represented as a collection of m sentences, $d_n=\{s_1,s_2,s_3,\ldots..s_m\}$. In the original DUC 2002 dataset, the documents are stored in sparse XML-shaped format show in Figure (3.2). Each document have multiple sentences. Each one of these sentences are stored in different files. This requires storing each sentence in each document based on the unique id that each document have.

<DOC>
<DOCNO> AP880911-0016 </DOCNO>
<FILEID>AP-NR-09-11-88 0423EDT</FILEID>
<FIRST>r i BC-HurricaneGilbert 09-11 0339</FIRST>
<SECOND>BC-Hurricane Gilbert,0348</SECOND>
<HEAD>Hurricane Gilbert Heads Toward Dominican Coast</HEAD>
<BYLINE>By RUDDY GONZALEZ</BYLINE>
<BYLINE>Associated Press Writer</BYLINE>
<DATELINE>SANTO DOMINGO, Dominican Republic (AP) </DATELINE>
<TEXT>
Hurricane Gilbert swept
toward the Dominican Republic Sunday, and the Civil Defense alerted its heavily populated south coast to prepare for high winds, heavy rains and high seas.
The storm was approaching from the southeast with sustained winds of 75 mph gusting to 92 mph.
``There is no need for alarm,'' Civil Defense Director Eugenio Cabral said in a television alert shortly before midnight Saturday.
Cabral said residents of the province of Barahona should closely follow Gilbert's movement. An estimated 100,000 people live in the province, including 70,000 in the city of Barahona, about 125 miles west of Santo Domingo.
Tropical Storm Gilbert formed in the eastern Caribbean and strengthened into a hurricane Saturday night. The National Hurricane Center in Miami reported its position at 2 a.m. Sunday at latitude 16.1 north, longitude 67.5 west, about 140 miles south of

Figure (3.2) XML-format of DUC2002 dataset

The process of reading the documents and their summarization are carried out separately. However, they are stored together in list of dictionaries data structure as in Figure (3.3). Each dictionary is holding

the document name, list carrying the document sentences, and list carrying the document summaries.

```
List of Dic = [{'doc_name':D1, 'docs':[s1, s2, ....],  'summary':[s1, s4, ....]},
               {'doc_name':D2, 'docs':[s1, s2, ....],  'summary':[s3, s6, ....]},
               {'doc_name':D3, 'docs':[s1, s2, ....],  'summary':[s1, s7, ....]},
               .
               .
               .
               .
               .]
```

Figure (3.3) The list of dictionaries to store the dataset

For the documents reading, the original corpus are texts files with extension ".S". These files are read each separately first. Notice that the content of the documents (i.e. document sentences) are stored separately in different files. Therefore, each new document is added to the list by using the document unique id as the document name. And any new sentences is checked first that is not exist in the list before adding it. Notice that the sentences are extracted from XML based template. The characters with '\n', '//', '-', '|', and '' are omitted. Any extra empty space is omitted too. For more detail, see algorithm (3.2).

After reading the documents and saves their sentences in the list of dictionary, the summaries are read. There is no extension for the summary files. However, they are read as normal text file. Like documents, the summaries are spared among many files. However, each sentence in the summary is provided with its original document id (e.g. document name). Looping through all the summaries and adding only the summaries with existing document names, automatically did not include the summaries with no documents sentences added before. Similarly to reading the original documents, the characters with '\n', '//', '-', '|', '' and extra empty spaces are omitted. For more detail, see algorithm (3.3)

After that, the extracted documents with no summaries found in the corpus are omitted too. The total number of the read documents and their summaries after data preprocessing are 510.

After reading the documents, these documents needs to be preprocessed in order to normalize the text that is included in it. The following are the steps involved in the preprocessing:

1. ***Sentence segmentation***: The beginning and end of each sentence are specified, thus the sentences are extracted from the set of documents separately.
2. ***Word tokenization:*** A token, such as a blank space, is used to split the words in every sentence.
3. ***Stop words removal:*** Possessives, articles, pronouns, conjunctions, prepositions, and other frequent words are examples of stop words. Furthermore, because stop words lack an emotion score, they have no effect on the sentiment of a phrase [106]. For these causes, they are eliminated from each sentence. The ROUGE package included

   a list of stop words that were employed in the suggested strategy [107].
4. ***Stemming:*** Stemming in Natural Language Processing, is defined as the steps followed to eliminate all prefixes and suffixes from a word in order to obtain the stems [108]. The major goal of the stemming preprocessing step is to unite the word style, which can take many different forms.

Algorithm(3-2) Document Reading and pre-processing
Input: topic folders T, document D
Output: list of dictionaries to store text of document doc
1: For each topic in T do //Read each folder t *2: Dict['topic']=t //get the topic name* *3: For each document d in D do //Read each file d in topic t* *4: Dict['name']=d['name'] //get the file name* *5: Text1=read(d) //read the content of d* *6: Text2=Text1.xml('text') //get the content of xml tag 'text'* *7: Text3=Text2.xml('s') //get the content of xml tag 's'* *8: Text4=Text3.remove("\n", "//" , "-" , "|" ,"") //clean text* *9: Text5=Text4.strip() // strip text* *10: Text6=Text5.stopwords_removal() // remove stopwords* *11: Text7=Text6.stemming // stemming* *12: Dict['doc']=Text7*

Algorithm (3-3) Summary Reading and pre-processing
Input: summary file SU
Output: list of dictionaries to store summary
1:For each summary file su in SU do //Read each file su *2: text1 = read(su) // Read the content of su* *3: text2 = text1.xml('text') //get the content of xml tag 'text'* *4: Text3=Text2.xml('s') //get the content of xml tag 's'* *5: Text4=Text3.remove("\n", "//" , "-" , "|" ,"") //clean text* *6: Text5 = text4.stip() //strip text* *7: if text5 is not in Dict && text5['name']in Dict: Dict['su']=text5* *8: Dict['summary']=Text5*

3.3.2 Similarity Analysis

The similarity between sentences must be calculated for the whole document, to search for related sentences. There are many methods for measuring similarity between sentences.

i. **Text Vectorization**

The vector-based word techniques are the most commonly employed in the field of text summarization. The sentences are represented as a vector of words in these approaches, and the similarity between them is measured by numeric value. The cosine similarity is the most commonly used in the field.

Assume $T= \{t_1, t_2,.., t_m\}$ be a set that contains all m different terms in the preprocessed document collection D. Each Sentence s_i can be represented as an m size vector $s_i = (w_{i1}, w_{i2},..., w_{im})$, $i= 1,2,...,n$, where n is the total number of sentences, and each element represents the TF-IDF weight of the word t_k inside the sentences s_i.

ii. TF-IDF bag of words similarity

The cosine similarity is used to measure the exact similarity between two vectors of sentences, where each vector contains the TF-IDF weight for each word in sentence for all document. The TF-IDF is calculated by equation (2.4) .

After calculated TF-IDF weight for each word in sentence, the cosine similarity between the sentences in all document are calculated by equation (2.5), it is equal to the cosine of the angle between them.

This cosine similarity is used in data science, for example, with text data to find out the extent to which texts match each other in searches. Many search engines such as Google use this scale to find matching words in texts. Which is similar to words in querying on websites, in addition to

its use in data retrieval processes, discovery of plagiarism and many other uses.

A mean vector $O = (o_1, o_2,..., o_m)$ containing the average weights of the m distinct terms from the TF-IDF weights can be used to quantitatively represent the main content or centre of D. This vector components are computed as equation (2.8)

iii. Graph Representation and Text Similarity

One of the common steps that makes processing the text data easier and more effective is representing the text as a graph. After the vectorization and the calculation of TF-IDF of the sentences in the document, these sentences are represented in the form of an indirect graph as shown in Figure (3.4). The text graph has nodes and edges. Each node represents a sentence. The edge represents a relationship between the nodes (i.e. relationship between the sentences in the text). Where the nodes are connected to each other's by the edges. These edges have a value (i.e. weight) to represent the strength between the sentences (i.e. sentences similarity). Here we use cosine similarity. For more details about similarity analysis, see Algorithm (3.4)

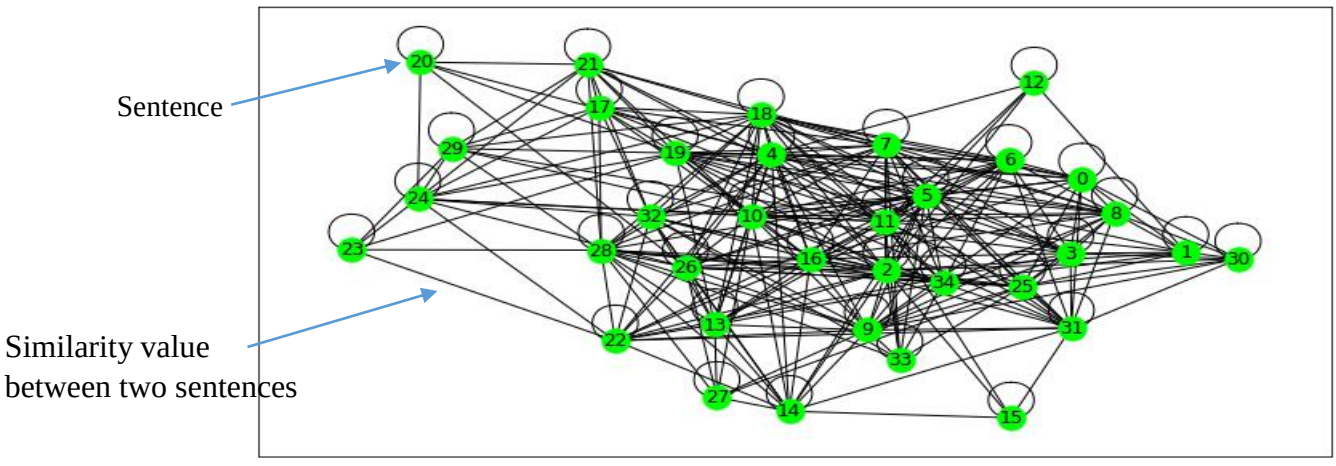


Figure (3.4) Graph representation of similarity between sentences

Algorithm(3-4) Similarity Analysis
Input: list of dictionaries that store data Dict['doc']
Output: syntactic_ similarity
1: all sentences=Dict['doc'] *2: for each sentences s in all sentences do* *3: for each word w in each sentence s do* *4: dt_matrix= TF-IDF(w)* *5: syntactic_ similarity =cosin similarity (dt_matrix,dt_matrix)* *5: similarity_graph = graph_network(similarity_matrix)*

3.3.3 Sentiment analysis

The purpose of Sentiment Analysis (SA) is to determine opinionated sentences with their orientation, as well as to compute their sentiment score. For this job:

First, choose a lexicon of sentiments. A sentiment lexicon is a dictionary in which each word is assigned one or more sentiment scores based on its context. In this thesis , Valence Aware Dictionary and sEntiment Reasoner (VADER) has been used(see section 2.4).

Second, the contextual polarity for each sentence is identified, because, it is effects on the polarity of the word in sentence .For more detail see section 2.4.

Finally, the sentiment score for each sentence is computed after assign sentiment score for each word from lexicon and determined the contextual polarity. The equation (2.6) show computing the sentiment score for each sentence in document.

After creating the similarity matrix computed by similarity measures as well as the sentiment similarity matrix computed, they are combined together to produce the summary that contains salient and

varied sentences that cover the content by equation (3.1). For additional details, see Figure (3.5).

$$\eta = \sum_{i=1}^{n}\sum_{j=1}^{n}(sim(s_{i,}s_j).senti(s_i,s_j)) \quad (3.1)$$

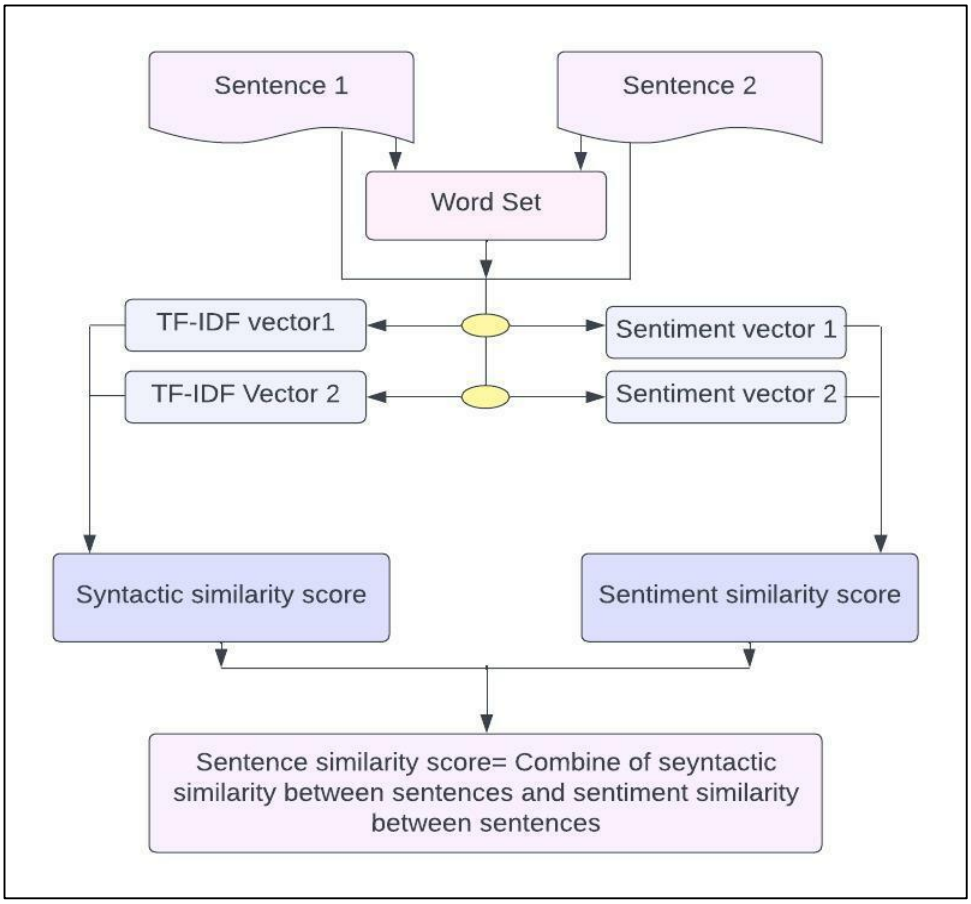


Figure (3.5) Sentence similarity computation

Where η is a heuristic information that combine the $sim(s_i,s_j)$ is syntactic similarity between sentence S_i and sentence s_j that shown in section (3.3.2), Algorithm(3.4) and the $senti(s_i,s_j)$ is sentiment similarity between sentence s_i and sentence s_j that shown in section (3.3.3).

3.3.4 The Unified objective Ant Colony optimization

The problem of sentiment-oriented text summarization can be solved in the form of an optimization problem. The proposed method uses the ACO algorithm to solve the text summarization problem. To solve the problem of the lack of heuristic information in the previous works, in this

thesis proposed a new heuristic function to come up with a new approach that can be described by the following steps:

i. Formulation of the optimization problem

In the proposed approach, the sentiment-oriented text summarization task has been drawn up like a multi-objective optimization problem. The aim in this problem is to produce a summary with the sentences from the document collection. These sentences should consider the minimum size of possible summarization. To achieve this goal, the following criteria are considered:

- Content coverage. The main content of the document set has to be covered in the generated summary by containing the most important sentences.
- Redundancy reduction. The generated summary must not contain similar sentences among them, so they must be avoided.
- Sentiment relevance. The sentiment score of the generated summary must be very close to the sentiment score of the document collection.

This problem involves optimizing three criteria at the same time: content coverage, sentiment relevance, and redundancy reduction.

The first criterion to be optimized in this work is in line with the content coverage criterion. Where the content of the document must be covered by the summary generated. This criterion is computed by calculating the cosine similarity function between each sentence in the summary generated s_i and the center of original document O. Therefore, this criterion must be maximized. The content coverage equation is illustrated in section (2.2.4), equation (2.1)

The second criterion to be optimized is related to the criterion of reducing redundancy between sentences in the summary generated *S*. To

achieve this criterion, the cosine similarity between each pair of sentences in generated summary *S*, s_i , s_j is calculated, and it must be minimized. The reducing redundancy equation is shown in section (2.2.4), equation (2.2).

As for the final criterion to be optimized, it is related to the criterion of the sentiment relevance. Measuring the difference between the sentiment score in the original document *D* and the summary generated *S*. The absolute maximum score of sentiment between the original document and the generated summary is 2, which means that the sentiment score between them are exactly the same. While when the sentiment score between them is equal to 0, this means that they are completely opposite in the sentiment score. The ranges of sentiment score in the interval [-1,1]. This objective must be maximized. The sentiment relevance equations are illustrated in section (2.2.4), equation (2.6),(2.7),(2.8).

Finally, after the three criteria are given, the unified-objective sentiment-oriented text summarizing problem can be written as follows:

$$\max \phi(X) = \{\Phi ConCov(X), \Phi \operatorname{Re} d \operatorname{Re} d(X), \Phi Sen \operatorname{Re} l(X)\} \qquad (3.2)$$

Where *ConCov(x)* is the Content coverage criterion, *RedRed(x)* is the Redundancy reduction criterion and *SenRel(x)* is the Sentiment relevance criterion.

ii. Unified-objective ant colony optimization algorithm

The steps of the algorithm of the unified-objective ant colony optimization (3.6) are as follows:

Firstly, loaded empty set of best solution *S*, this set store paths of the ants. Initialize parameter of ACO algorithm that involve Ant_size: number of ant, *p*: evaporation rate, t_0: initial pheromone trail, α: weight of

pheromone information, β: weight of heuristic information, iteration: the number of construction times, these steps are carried out (line 1).

Secondly, after the ant choose the initial position from collection of sentences is randomly. Each ant constructs a solution for the input document by construction solution equation that showed in equation (2.15) that use two criteria (heuristic information η, which it is input to algorithm that illustrated in equation (3.1) and pheromone information *t*). Where compute probability (construction solution equation) for each node, these probabilities entire on roulette wheel to choose next sentences by stochastic behavior. These steps are carried out (lines 2-6).

After each ant constructs the solutions (generates all paths), the algorithm chooses the best path that have biggest similarity value (cost of the path) between nodes. These steps are carried out (line 7 and line 8).

Thirdly, the pheromone for all paths is evaporated to avoid stagnation for the algorithm by evaporate equation, showed in equation (2.16).Show in line 9 in algorithm (3.5).

After that, the path that have biggest similarity summarize by summary rate. Show in line 10 in algorithm (3.5).

Then, reward the best path by increasing the pheromone concentration of the path by deposit equation**,** showed in equations (2.17),(2.18).

Where *Q* in equation (2.18) is a positive parameter that combines three criteria: content coverage (equation 2.7), redundancy reduction (equation 2.10), and sentiment similarity (equation 2.11). These criteria apply on the best path after summarize it by omitting the nodes that have the minimum pheromone. These three criteria combine in the equation (3.2). These steps are carried out (lines 11-19).

These steps are repeated until it reaches to stop condition. Upon finalization of the loop, the return path is filtered by omitting the nodes that have less amount of pheromone (line 20).

Algorithm(3-5) Ant colony optimization

Input: Dis //Distance matrix η equation (3.1), summary rate.

Output: set of best solution S

1:initial: Ant_size: number of ant, p: evaporation rate, t_0: initial pheromone trail, alpha : weight of pheromone information, beta : weight of heuristic information, Iteration: the number of construction times

2: For i in {1,2,…,Iterations} do

// generate the possible paths//

3: For each ant in {1,2,…, Ants_size} do

4: For each d in Dis do

5: *track_norm =* $P_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{i \in J_i^k} [\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}$

6: index = random(Dis based on track_norm) //roulette selection

// calculate the maximize distance of current path//

7: If the distance of current path > the distance of the best path

8: best_path= current_path

9: pheromones= t_0(1- p) //Evaporation*

10: summarization best_path= best_path summary rate*

// calculate content coverage//

11: cos_sim = cosine_similarity(summarization best_path, centre of document)

// calculate the redundancy reduction//

12: redun_reduc = 0

13: For each node x in summarization best _path-1 do

14: For each node y in range (x+1, summarization best _path)do

15: redun_reduc+=cosine_similarity(x,y)

// calculate sentiment relevance//

16: sent_sim=2-abs(polarity_scores[document] - polarity_scores[: summarization best_path])

//unified three criteria//

17: equ_(3.2) = max(1/redun_reduc, cos_sim, sent_sim)

//spread pheromones//

18: For each node d in the best_path do

19: pheromone = t_0 + equ_(3.2)/(distance of current_path)

// Return the best path(summary)//

20: S= summarization best_path

Chapter Four

The Experimental Results and Discussion

The Experimental Results and Discussion

4.1 Introduction

In this chapter, the obtained results will be displayed and discussed. For the matter of text summarization problem, there are different methods to show the performance of text summarization models. In general, the majority of extractive text summarization model evaluation are based on the presence of human generated texts. In other words, the model will be evaluated by giving it a group of complete documents (corpus). These documents are summarized by humans by selecting the most important and representative sentences manually from the document. The human selected summarization will be compared with the summarization of the model in order to determine its performance quality.

This is done through different evaluation metrics that are explained in details in chapter two. In this chapter the results are showed directly by using these metrics, with no further explaining, considering they are explained previously.

4.2 Description of DUC2002 Dataset

There are numerous public datasets available to evaluate summarization system. The Document Understanding Conference (DUC) datasets has been employed in this thesis. The National Institute of Standards and Technology (NIST) provides these datasets, which are the most commonly and widely used datasets for text summarization. The DUC dataset is available as part of the DUC conference's summarizing shared task. The most recent DUC challenge took place in 2007. DUC datasets from 2001 to 2007 are available on the DUC website. Each dataset consists of both documents (news articles) and their manually generated summaries. Table (4.1) describes the DUC dataset.

Table (4.1).DUC dataset for text summarization

	Dataset Name	Number of Document	Language	Domain
1	DUC 2001	600	English	News
2	DUC 2002	600	English	News
3	DUC 2003	600، 750	English	News
4	DUC 2004	1000	Arabic، English	News
5	DUC 2005	160	English	News
6	DUC 2006	1.250	English	News
7	DUC 2007	250	English	News

DUC2002 dataset has been employed in this thesis. This dataset consists both documents (news articles) and their summaries. Table (4.2) presents several characteristics of these datasets.

Table (4.2) characteristics of DUC2002 dataset

Description of DUC2002	Value
No. of topics	59
Average no. of documents	10
Average no. of sentences for document	35
Average no. of total terms	5.454
Average no. of unique terms	1.056
Summary length constrain	400

4.3 Developing Environment

For the purpose of developing and testing the project, Python 3 programming language is selected due to its easiness in coding, and the availability of free and wide libraries and packages. Furthermore, Google Colaboratory is selected as the developing environment due to its higher performance comparing with the regular and commercial personal computers. Since it gives a high level of CPU and GPU resources.

The used laibraries in this work are *glob*, *os*, *TfidfVectorizer*, *pandas*, *numpy*, *nltk*, *networkx*, *matplotlib.pyplot*, *math*, *collections*, *rouge*, and *statistics.*

4.4 Parameters settings

There are some parameters that need to be set in this thesis. In this section, the effects of efficiency caused by the parameters of SO-ACO are analyzed. On the other hand, through the intensive experimental study, the influence of changing the parameters has been studied experimentally using different settings of 10 independent executions of the algorith ms. The maximum number of tour constructions is 40(no improvement over this iteration number). The number of ants Ant _ Size , evaporation rate ρ, initial pheromone trail τ, and the weights of pheromone and heuristic formation (α, β) are varied among candidate values except one parameter keeps unchanged. The ranges of these parameters are listed in Table (4.3).

Table (4.3) range of the SO-ACO parameters

parameters	Range
N	{10,12,15,20,40,50}
α , B	{(0.5,1),(1,0.5),(0.5,1.5),(1.5,0.5),(0.5,2),(2,0.5)
P	{0,01 , 0.03 , 0.05 , 0.07 , 0.09 , 0.1}
$\mathcal{T}$	(0.001 , 0.005 , 0.01 , 0.015 , 0.02 , 0.03}
$\mathcal{Q}$	Max or average

The algorithm performance is vibrating with the ants' size. With a more little value of ants' size, solution goodness is often decreased. Whereas Ant _ Size is exacerbated, additional candidate solutions can be erected at each cycle, and the best solution obtained from each run is customarily preponderant. This may be due to the collaboration between ants. Consequently, when the number of ants equals 10, the balance between maximum and minimum number of ants is adapted to overcome this gap.

The performance of the algorithm is changing with α and β that represent the weights of pheromone and heuristic information. It is

observed that there is no noticeable change when the value of the α parameter is assigned from zero to 0.5. In contrast, when the β parameter is assigned a value more than 1.5, it results in stagnation straight away. The best values are obtained when α and β parameters are 0.5, and 1.5 respectively.

The performance of the algorithm is changing with ρ (Pheromone evaporation constant).Particularly, the performance of the algorithm is the worst with increasing the value of ρ the pheromone trails on arcs which are not less reinforced decrease faster. As a result, the search converges earlier around the best tours.

The performance of the algorithm is changing with τ (Pheromone initialization constant) .performance of the algorithm is the worst when τ equal 0.001 and 0.005. The corresponding algorithm results are not very satisfactory when τ is too small or too large.

The performance of the algorithm is almost identical and there are no significant differences with Q (Pheromone update constant) when q is equal of the Max or Average of the three criteria that showed in equation (3.2).

Conspicuously, the maximal performance of the SO-ACO algorithm is obtained when the parameters' values are configured as in Table (4.4)

Table (4.4) parameters of SO-ACO

Parameter	Value	Description
N	20	The number of ants
α	0.5	Weight for pheromone level
B	1.5	Weight for heuristic information
P	0.05	Pheromone evaporation constant
$\mathcal{T}$	0.01	Pheromone initialization constant
$\mathcal{Q}$	Max , average	Pheromone update constant
Iteration	40	Number of iteration
Trial	10	Number of trials for evaluation

4.5 Evaluation metrics and Results

In this work, there are several metrics have been used to assess the quality of the generated summaries. These metrics are shown below:

4.5.1 Rouge family

The Rouge family consist of different types of evaluation metrics as showed in chapter two. For the purpose of this work, three Rouge metrics are selected, which are Rouge-1, Rouge-2, and Rouge-L. for each one of them, the Recall, Precision, and F1-score are calculated (as showed in chapter two) for each document. The mean, median, and standard deviation are calculated for measure in each of the Rouge metrics. Tables (4.5) and (4.6) show these results.

Table (4.5) Rouge results where Q=Average

Rouge		Mean	Median	SD
Rouge 1	F1-score	0.6010	0.6395	0.2057
	Precision	0.6218	0.6688	0.2113
	Recall	0.5870	0.6248	0.2049
Rouge 2	F1-score	0.5294	0.5637	0.1977
	Precision	0.5546	0.5823	0.2023
	Recall	0.5180	0.5494	0.1976
Rouge-L	F1-score	0.5908	0.6326	0.2046
	Precision	0.6143	0.6614	0.2101
	Recall	0.5802	0.6162	0.2038

Table (4.6) Rouge results where Q=Max

Rouge		Mean	Median	SD
Rouge 1	F1-score	0.5970	0.6321	0.1932
	Precision	0.6143	0.6563	0.2023
	Recall	0.5751	0.6181	0.1972
Rouge 2	F1-score	0.5178	0.5597	0.1901
	Precision	0.5476	0.5783	0.1978
	Recall	0.5070	0.5378	0.1902
Rouge-L	F1-score	0.5867	0.6210	0.1959
	Precision	0.6076	0.6521	0.2088
	Recall	0.5763	0.6075	0.1977

Table (4.5) shows the Rouge-1, Rouge2, and Rouge-L in each of the F1-score, Precision, and Recall. Each one of them calculate the mean, median, and standard deviation. These results are derived from calculating the average of the three criteria's mentioned previously. While Table (4.6) shows the same results, but by using the max value for thee three criteria's. This thesis will depend on the Table (4.5) to extract other results.

The majority of researches use the mean of F1-score as their evaluation metric. Considering that, the results of our work is 0.6010 Rouge-1, 0.5294 Rouge-2, and 0.5908 Rouge-L.

Figures (4.1), (4.2), and (4.3) shows the mean of Rouge-1, Rouge-2, and Rouge-L values for 59 topics. The histogram for Rouge 1,Rouge 2, and Rouge L score in Figure (4.1) , (4.2), and (4.3) shows approximate asymmetrical distribution (left skewed).

The boxplot for Rouge 1, Rouge2, and Rouge L scores show only one outlier.

As shown in histogram of Figure (4.1), the majority of the results have the Rouge-1 value from 5 to 8. However, there are some documents with the value 1 (the lowest value), and some documents with value 9 (the highest value). On the other hand, the boxplot of Figure (4.1) shows the whole documents have Rouge-1 value from 5 to 9, with one outlier with value of 1.

Figure (4.2), the histogram shows the majority of the results have the Rouge-2 value from 6 to 8. However, there are some documents with the value 1 (the lowest value), and some documents with value 9 (the highest value). On the other hand, the boxplot of Figure (4.2) shows the whole documents have Rouge-1 value from 5 to 9, with one outlier with value of 1.

Figure (4.3), shows the results of Rouge-L. The histogram shows the majority of the results have the Rouge-L value from 6 to 8. However, there are some documents with the value 1 (the lowest value), and some documents with value 9 (the highest value). On the other hand, the boxplot of Figure (4.3) shows the whole documents have Rouge-L value from 5 to 9, with one outlier with value of 1.

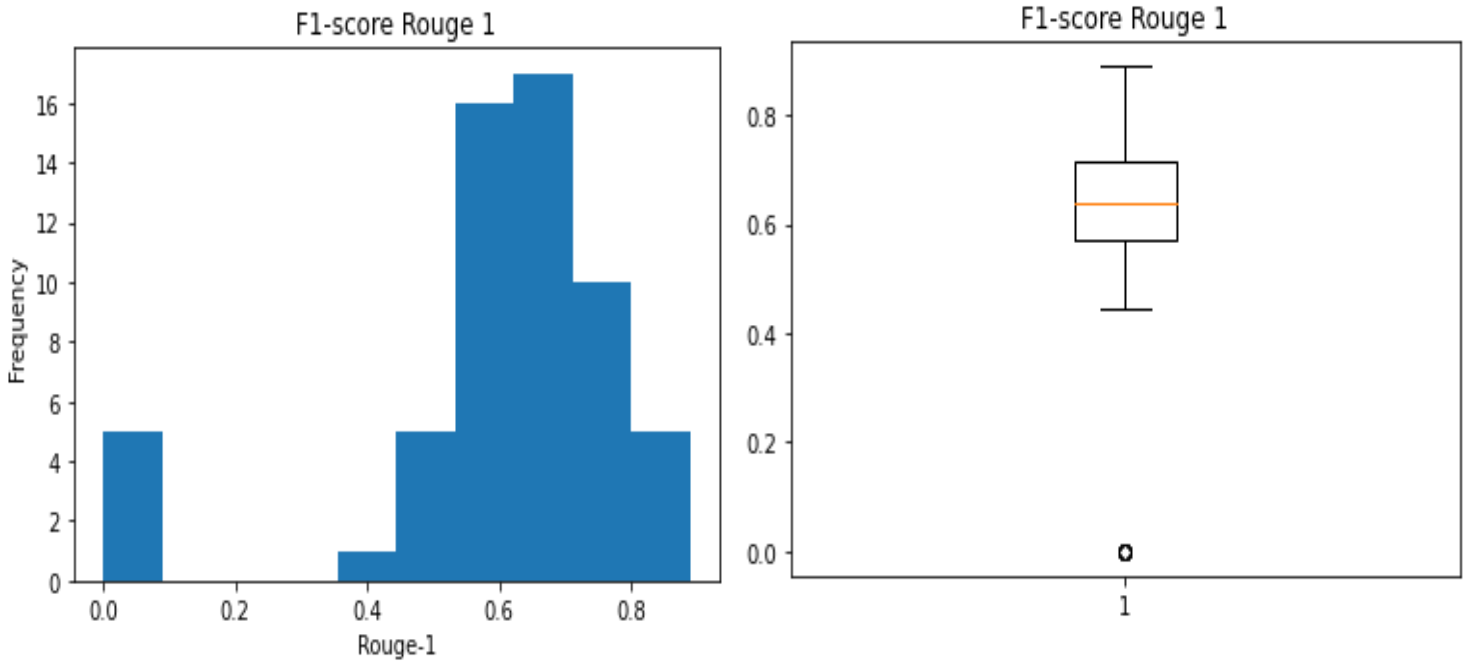


Figure (4.1) Histogram and boxplot for Rouge-1 score

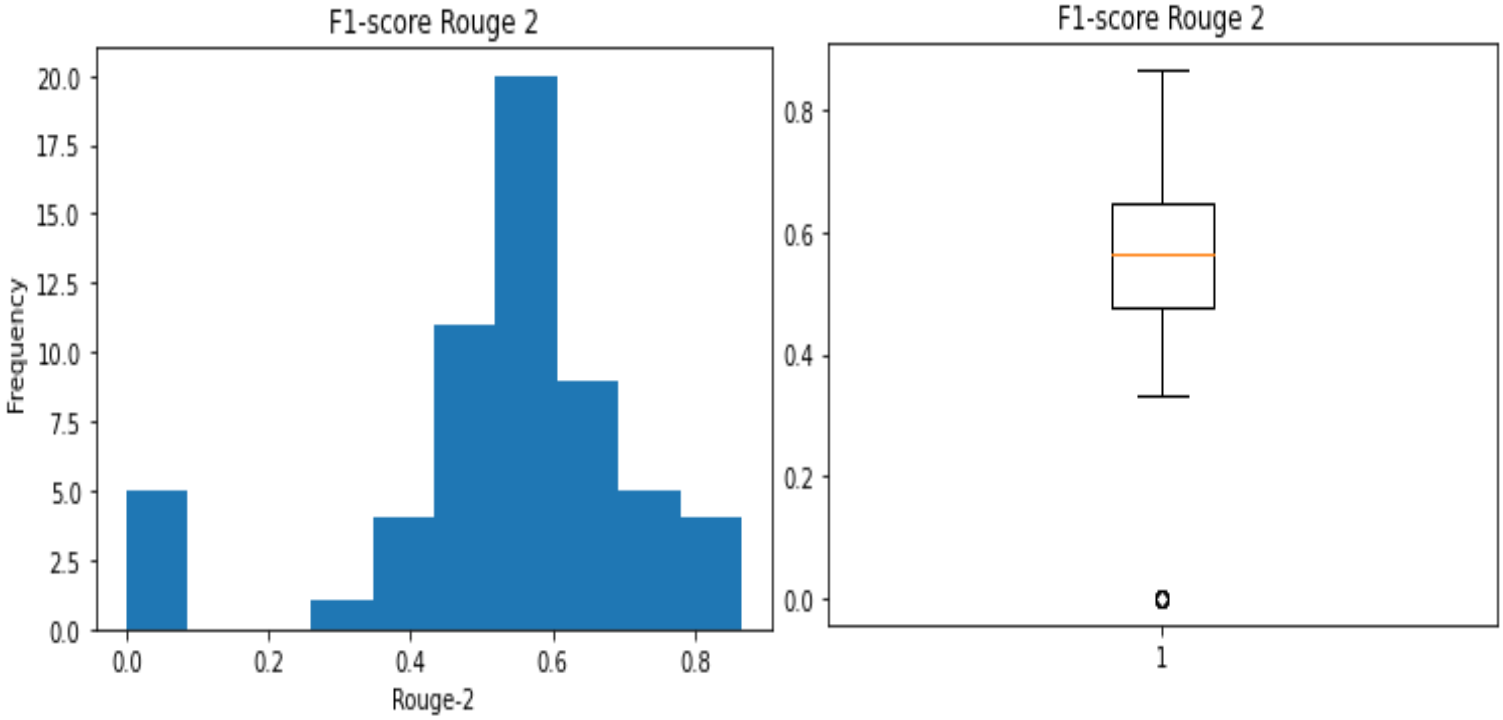


Figure (4.2) Histogram and boxplot for Rouge-2 score

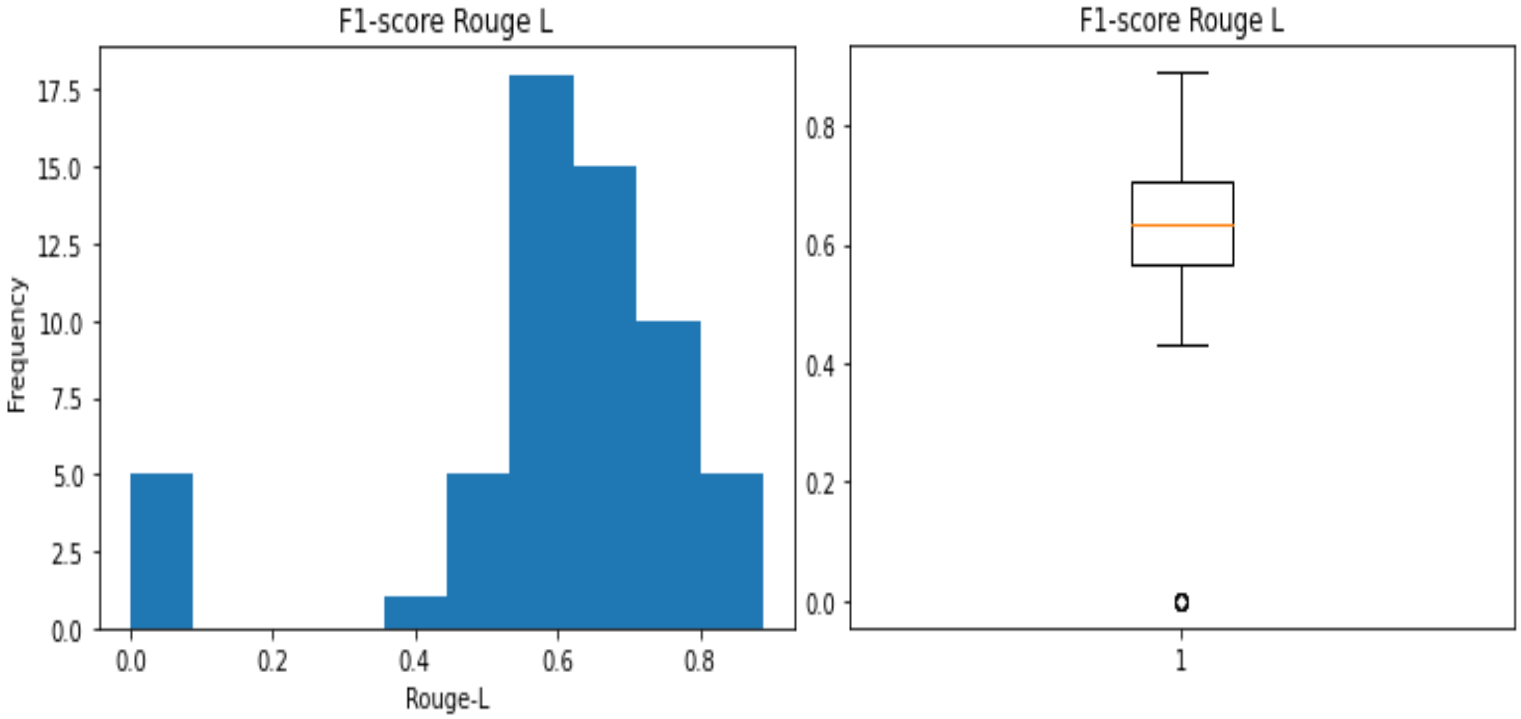


Figure (4.3) Histogram and boxplot for Rouge-L score

4.5.2 Evaluation the proposed system Depending on the three criteria

Another measure is considered for the model evaluation. This measure is considered to ensure that the model performance is high from different aspects and by using different metrics. This measure is constructed from three metrics, as shown in section (2.2.4) of this work. These three metrics are used in some works to evaluate the text summarization models. However, they are not popular as Rouge metrics.

i. Content coverage

The content coverage is responsible for calculating how much the content is covered in the summarization. This is done by using the cosine similarity between the summarized sentences and the center of the document. Which is calculated by using the mean vector of the document. This is shown in equation (2.1).

ii. Redundancy Reduction

The redundancy reduction is responsible for calculating how much each summarization reduced the original text. The more the text reduced, the more efficient the model. Equation (2.4) shows how the redundancy reduction is calculated.

iii. Sentiment relevant

The sentiment relevant between the documents and their summarizations. Equation (2.6) shows how the sentiment relevant is calculated.

The three criteria results are as displayed in Table (4.7):

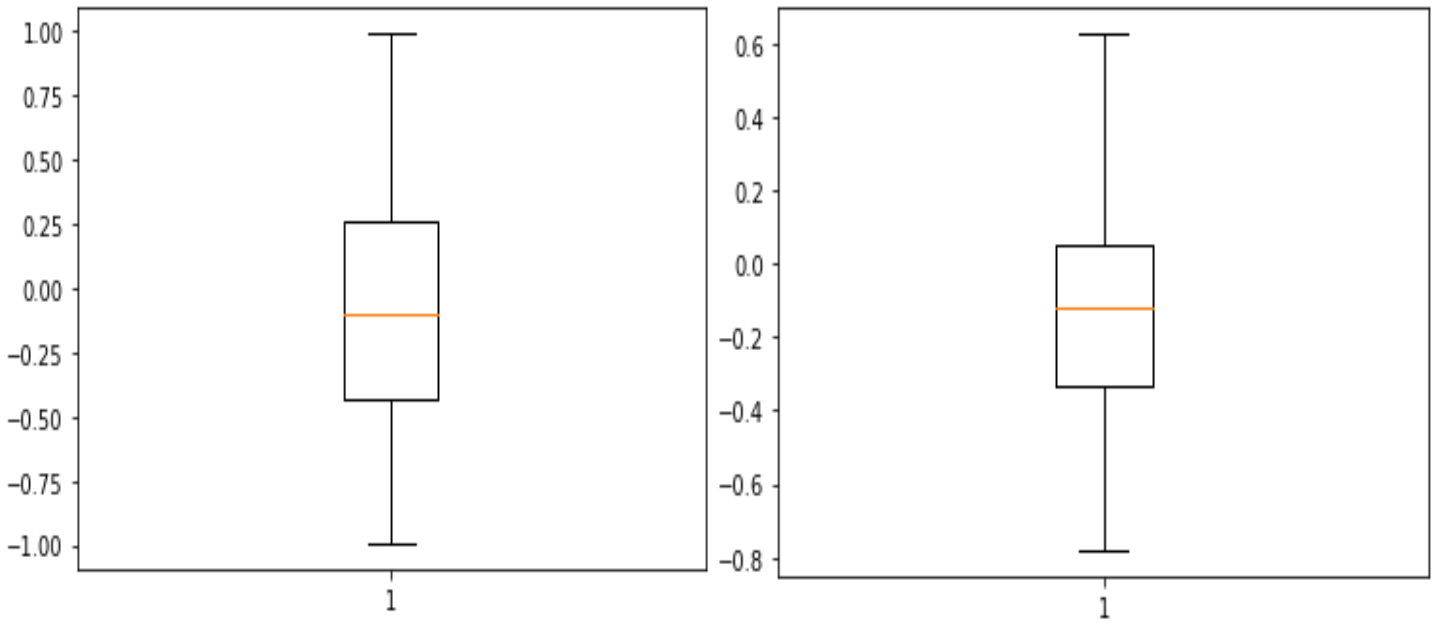
Table (4.7) Three criteria's results

	Redundancy reduction	**Sentiment relevant**	**Content coverage**
Mean	0.627	1.868	0.76
Median	0.417	1.896	0.736
Standard deviation	0.51	0.122	0.169

4.5.3 Pearson correlation coefficient

Concerning the scores of sentiment relevance, it has not been possible to compare sentiment relevance scores with other methodologies because none of them reveal their results. As a result, it has been compared to the ideal case. When the Pearson correlation coefficient is equal to +1, we have reached the ideal case. The Pearson correlation coefficient for the 59 topics from the DUC2002 datasets is found to be r = 0.8603782, indicating a strong and robust linear positive correlation, between both the sentiment scores calculated for the generated summaries and the sentiment scores calculated for their relevant topics.

Figure (4.4) shows a comparison of the boxplots for both the sentiment scores for the generated summaries and the sentiment scores for the corresponding topics.



Topic sentiment score Summary sentiment score

Figure (4.4) Boxplots for the sentiment scores of the summaries generated and for sentiment scors of the corresponding topics

Figure (4.5) show the scatter plot between sentiments scores calculated for generated summary and sentiment scores of the corresponding topics. It shows the strong positive correlation between the two variables that explain above.

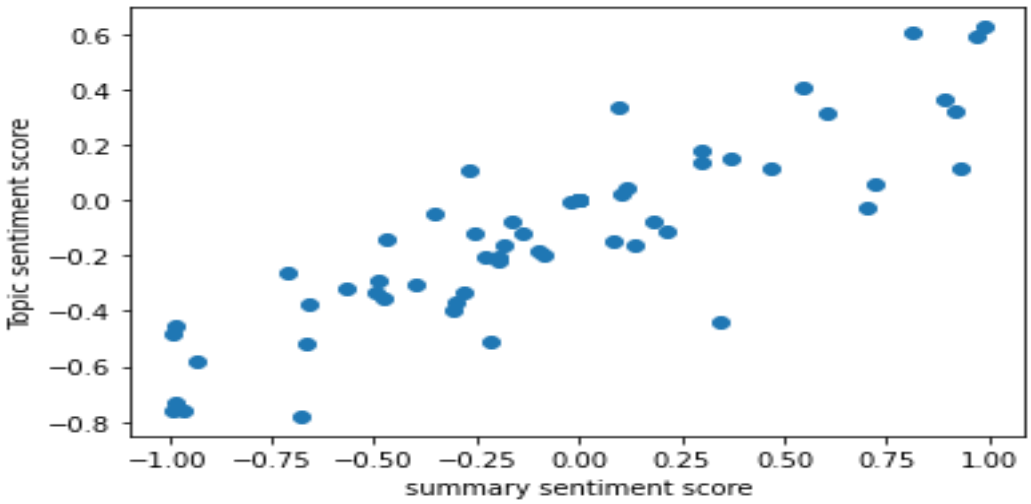


Figure (4.5) scatter plot of the sentiment score of summaries generatied and sentiment score of the corresponding topics

4.6 Comparison with other approaches

The results gained by the SO-ACO algorithm are compared to those obtained by other methodologies given by other researchers in the scientific literature (see section 1.7). Table (4.7) presents the results for the 59 topics from the DUC2002 datasets. The values provided for the SO-ACO method are the mean F-measure for Roge-1and Rouge-2, similar to the values reported by the other approaches for the same dataset.

Table (4.8) comparison between SO-ACO algorithm and other approaches for Rouge 1 and Rouge 2 scores.

References	Rouge1	Rouge 2
SO-ACO	**0.6010**	**0.5294**
[11]	0.553	0.342
[17]	0.4908	0.2309
[18]	0.4781	0.0869
[20]	0.48423	0.22471

It is noticeable that the SO-ACO algorithm outperforms all other approaches in terms of mean F-measure ROUGE scores for the ten independent runs.

Relative improvement of SO-ACO over the other approach are also reported. The average percentage for relative improvement present in Table (4.9) are + 20.23 % for Rouge-1 and +207.21% for Rouge-2 .The relative improvement of SO-ACO over another approach X was computed as follows:

$$Relative\ Improvment = \frac{score(SO-ACO)-score(X)}{score(x)} * 100 \quad (4.1)$$

Table (4.9) Relative improvement of SO-ACO algorithm over other systems

References	Rouge1	Rouge 2
[11]	+ 8.67 %	+ 54.79 %
[17]	+ 22.45%	+ 129.27%
[18]	+ 25.70 %	+ 509.20%
[20]	+ 24.12 %	+135.60 %
Average	+ 20.23 %	+ 207.21 %

4.7 Discussion of the result

The key purpose of the SO-ACO model is to improve automatic text summarization by formulating a new heuristic function that combines syntactic similarity with semantic similarity. In which the ACO is used to formulate the documents as nodes. The most representative sentences of the document will be the central sentences. Which have the most connections to the other sentences. Therefore, it calculate the TF-IDF and

sentiment similarity between the sentences as the pheromone of the ACO between sentences. In addition, it takes into account the three criteria: content coverage, redundancy reduction, and sentiment relevance that are used in the update pheromone equation.

The experimental results showed the superiority of the proposed model compared to other studies using the Rouge metric: Rouge1 and Rouge2 (other authors did not consider Rouge-L for the evaluation of their works), shown in Table (4.8). The results for all the generated summaries were compared with the summaries generated by humans for the DUC2002 dataset using Rouge-1, Rouge-2, and Rouge-L that illustrate in Table (4.5) and (4.6). Their histograms showed an approximate asymmetric distribution (skewed to the left) as showed in Figures (4.1), (4.2), (4.3).

The three criteria shown in Table (4.7) were also calculated to provide additional insight on the performance of the proposed system.

The Pearson correlation coefficient for sentiment score shows a strong linear positive correlation between the generated summary and the 59 topics in the original document, as shown in the scatter plot in Figure (4.5).

Chapter Five

Conclusions and Future Work

5.1 Conclusions

The most important characteristics that have been discovered through the implementation of the proposed methodology and the discussion of its results are as follows:

1. The proposed methodology proved its effectiveness in determining the sentences most closely related to each other by formulating a heuristic equation concerned with sentiment analysis and similarity analysis.
2. The proposed methodology proved effectiveness in generating a summary that concerned with the following aspects:
   i. Analysing the author's opinion on documents and his sentiment orientation by delving into the field of sentiment analysis and opinion exploration.
   ii. Reducing the redundancy of sentences in the produced summary and making it as minimal as possible using measures that reduce the redundancy.
   iii. Covering the main content of the set of documents by including the summary produced on the most important sentences in the original document.

   As these aspects are maximized in a simultaneous manner
3 The recorded results of the proposed methodology indicated its superiority compared to the methods used in the previous literature using the Rouge Scale.
4 The score of sentiment relevance is evaluated using the Pearson coefficient. The results indicated that there is a strong linear positive correlation (0.8603782) between the score of sentiment for the original documents and the score of sentiment for the produced summaries.

5.2 Suggestion for Future Research

In this section some future directions are suggested as an extension of the proposed system in this thesis:

1. Applying the proposed model to other dataset such as TAC 2011 MultiLing Pilot that Supports Arabic and English languages, and compare the obtained results with other studies used these datasets.
2. Use pairs of word to TF-IDF (bigram, trigram etc.) instead of a single word count to minimize the matrix of TF-IDF, and therefore minimize the time of calculation.
3. Taking into account the information provided by the user by developing a sentiment-based query-oriented text-summarization approach.
4. Making this approach as a tool in e-learning platforms in order to summarize written texts for students.
5. Using semantic similarity instead of sentiment similarity.

References

References

[1] W. S. El-Kassas, C. R. Salama, A. A. Rafea, and H. K. Mohamed, "Automatic text summarization: A comprehensive survey," *Expert Syst. Appl.*, vol. 165, p. 113679, 2021, doi: 10.1016/j.eswa.2020.113679.

[2] J.-M. Torres-Moreno, *Automatic Text Summarization*. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2014.

[3] N. Moratanch and S. Chitrakala, "A survey on extractive text summarization," in *2017 International Conference on Computer, Communication and Signal Processing (ICCCSP)*, Jan. 2017, pp. 1–6, doi: 10.1109/ICCCSP.2017.7944061.

[4] H. Lin and V. Ng, "Abstractive Summarization: A Survey of the State of the Art," *Proc. AAAI Conf. Artif. Intell.*, vol. 33, pp. 9815–9822, Jul. 2019, doi: 10.1609/aaai.v33i01.33019815.

[5] S. A. Curtis, "The classification of greedy algorithms," *Sci. Comput. Program.*, vol. 49, no. 1–3, pp. 125–157, Dec. 2003, doi: 10.1016/j.scico.2003.09.001.

[6] F. Archetti and F. Schoen, "A survey on the global optimization problem: General theory and computational approaches," *Ann. Oper. Res.*, vol. 1, no. 2, pp. 87–110, Jun. 1984, doi: 10.1007/BF01876141.

[7] M. Abdel-Basset, L. Abdel-Fatah, and A. K. Sangaiah, "Metaheuristic Algorithms: A Comprehensive Review," in *Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications*, Elsevier, 2018, pp. 185–231.

[8] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE Comput. Intell. Mag.*, vol. 1, no. 4, pp. 28–39, Nov. 2006, doi: 10.1109/MCI.2006.329691.

[9] M. Dorigo and T. Stützle, *Optimization*. .

[10] J. M. Sanchez-Gomez, M. A. Vega-Rodríguez, and C. J. Pérez, "Sentiment-oriented query-focused text summarization addressed with a multi-objective optimization approach," *Appl. Soft Comput.*, vol. 113, p. 107915, Dec. 2021, doi: 10.1016/j.asoc.2021.107915.

[11] J. M. Sanchez-Gomez, M. A. Vega-Rodríguez, and C. J. Pérez, "A decomposition-based multi-objective optimization approach for

extractive multi-document text summarization," *Appl. Soft Comput. J.*, vol. 91, p. 106231, 2020, doi: 10.1016/j.asoc.2020.106231.

[12] B. Gunel, C. Zhu, M. Zeng, and X. Huang, "Mind The Facts: Knowledge-Boosted Coherent Abstractive Text Summarization," no. NeurIPS, pp. 1–7, 2020, [Online]. Available: http://arxiv.org/abs/2006.15435.

[13] Z. Li, Z. Peng, S. Tang, C. Zhang, and H. Ma, "Text Summarization Method Based on Double Attention Pointer Network," *IEEE Access*, vol. 8, pp. 11279–11288, 2020, doi: 10.1109/ACCESS.2020.2965575.

[14] D. Miller, "Leveraging BERT for Extractive Text Summarization on Lectures," 2019, [Online]. Available: http://arxiv.org/abs/1906.04165.

[15] Lucky and A. S. Girsang, "Multi-objective ant colony optimization for automatic social media comments summarization," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 3, pp. 400–408, 2019, doi: 10.14569/IJACSA.2019.0100352.

[16] R. Elbarougy, G. Behery, and A. El Khatib, "Extractive Arabic Text Summarization Using Modified PageRank Algorithm," *Egypt. Informatics J.*, vol. 21, no. 2, pp. 73–81, 2020, doi: 10.1016/j.eij.2019.11.001.

[17] R. M. Alguliyev, R. M. Aliguliyev, N. R. Isazade, A. Abdi, and N. Idris, "COSUM: Text summarization based on clustering and optimization," *Expert Syst.*, vol. 36, no. 1, p. e12340, Feb. 2019, doi: 10.1111/exsy.12340.

[18] A. Abdi, S. M. Shamsuddin, S. Hasan, and J. Piran, "Automatic sentiment-oriented summarization of multi-documents using soft computing," *Soft Comput.*, vol. 23, no. 20, pp. 10551–10568, Oct. 2019, doi: 10.1007/s00500-018-3653-4.

[19] A. B. Al-Saleh, M. El, and B. Menai, "Ant Colony System for Multi-Document Summarization," *Coling*, pp. 734–744, 2018.

[20] E. Vázquez, R. Arnulfo García-Hernández, and Y. Ledeneva, "Sentence features relevance for extractive text summarization using genetic algorithms," *J. Intell. Fuzzy Syst.*, vol. 35, no. 1, pp. 353–365, 2018, doi: 10.3233/JIFS-169594.

[21] M. A. Mosa, A. Hamouda, and M. Marei, "Ant colony heuristic for user-contributed comments summarization," *Knowledge-Based Syst.*, vol. 118, pp. 105–114, 2017, doi:

10.1016/j.knosys.2016.11.009.

[22] O. F. Hassan, “Text Summarization Using Ant Colony,” no. February. 2015.

[23] H. P. Luhn, “The Automatic Creation of Literature Abstracts,” *IBM J. Res. Dev.*, vol. 2, no. 2, pp. 159–165, Apr. 1958, doi: 10.1147/rd.22.0159.

[24] K. Spärck Jones, “Automatic summarising: The state of the art,” *Inf. Process. Manag.*, vol. 43, no. 6, pp. 1449–1481, Nov. 2007, doi: 10.1016/j.ipm.2007.03.009.

[25] J. Goldstein, V. Mittal, J. Carbonell, and M. Kantrowitz, “Multi-document summarization by sentence extraction,” in *NAACL-ANLP 2000 Workshop on Automatic summarization* -, 2000, vol. 4, pp. 40–48, doi: 10.3115/1117575.1117580.

[26] C. C. Yang and F. L. Wang, “Hierarchical summarization of large documents,” *J. Am. Soc. Inf. Sci. Technol.*, vol. 59, no. 6, pp. 887–902, Apr. 2008, doi: 10.1002/asi.20781.

[27] S. Harabagiu and F. Lacatusu, “Using topic themes for multi-document summarization,” *ACM Trans. Inf. Syst.*, vol. 28, no. 3, pp. 1–47, Jun. 2010, doi: 10.1145/1777432.1777436.

[28] L. Huang, Y. He, F. Wei, and W. Li, “Modeling Document Summarization as Multi-objective Optimization,” in *2010 Third International Symposium on Intelligent Information Technology and Security Informatics*, Apr. 2010, pp. 382–386, doi: 10.1109/IITSI.2010.80.

[29] T. Vodolazova, E. Lloret, R. Muñoz, and M. Palomar, “Extractive Text Summarization: Can We Use the Same Techniques for Any Text?,” 2013, pp. 164–175.

[30] S. Muresan, E. Tzoukermann, and J. L. Klavans, “Combining linguistic and machine learning techniques for email summarization,” in *Proceedings of the 2001 workshop on Computational Natural Language Learning - ConLL ’01*, 2001, vol. 7, pp. 1–8, doi: 10.3115/1117822.1117837.

[31] N. Alampalli Ramu, M. S. Bandarupalli, M. S. S. Nekkanti, and G. Ramesh, “Summarization of Research Publications Using Automatic Extraction,” 2020, pp. 1–10.

[32] and G. S. L. Gupta, Vishal, “A survey of text summarization extractive techniques,” *ournal Emerg. Technol. web Intell.*, vol. 2,

no. 3, pp. 258–268, 2010.

[33] D. R. Radev, E. Hovy, and K. McKeown, “Introduction to the Special Issue on Summarization,” *Comput. Linguist.*, vol. 28, no. 4, pp. 399–408, Dec. 2002, doi: 10.1162/089120102762671927.

[34] K. Sarkar, “Syntactic trimming of extracted sentences for improving extractive multi-document summarization,” *J. Comput.*, vol. 2, pp. 177–184.

[35] M. A. Mosa, A. Hamouda, and M. Marei, “Graph coloring and ACO based summarization for social networks,” *Expert Syst. Appl.*, vol. 74, pp. 115–126, 2017, doi: 10.1016/j.eswa.2017.01.010.

[36] M. Gambhir and V. Gupta, “Recent automatic text summarization techniques: a survey,” *Artif. Intell. Rev.*, vol. 47, no. 1, pp. 1–66, 2017, doi: 10.1007/s10462-016-9475-9.

[37] C. Y. Liu, M. S. Chen, and C. Y. Tseng, “IncreSTS: Towards Real-Time Incremental Short Text Summarization on Comment Streams from Social Network Services,” *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 11, pp. 2986–3000, 2015, doi: 10.1109/TKDE.2015.2405553.

[38] M. Al-Dhelaan, “StarSum: A simple Star Graph for multi-document summarization,” *SIGIR 2015 - Proc. 38th Int. ACM SIGIR Conf. Res. Dev. Inf. Retr.*, pp. 715–718, 2015, doi: 10.1145/2766462.2767790.

[39] Y. Ouyang, W. Li, S. Li, and Q. Lu, “Applying regression models to query-focused multi-document summarization,” *Inf. Process. Manag.*, vol. 47, no. 2, pp. 227–237, 2011, doi: 10.1016/j.ipm.2010.03.005.

[40] M. A. Fattah and F. Ren, “GA, MR, FFNN, PNN and GMM based models for automatic text summarization,” *Comput. Speech Lang.*, vol. 23, no. 1, pp. 126–144, Jan. 2009, doi: 10.1016/j.csl.2008.04.002.

[41] K. Riedhammer, B. Favre, and D. Hakkani-Tür, “Long story short - Global unsupervised models for keyphrase based meeting summarization,” *Speech Commun.*, vol. 52, no. 10, pp. 801–815, 2010, doi: 10.1016/j.specom.2010.06.002.

[42] W. Song, L. Cheon Choi, S. Cheol Park, and X. Feng Ding, “Fuzzy evolutionary optimization modeling and its applications to unsupervised categorization and extractive summarization,” *Expert Syst. Appl.*, vol. 38, no. 8, pp. 9112–9121, Aug. 2011, doi:

10.1016/j.eswa.2010.12.102.

[43] Y. Chali and S. A. Hasan, “Query-focused multi-document summarization: Automatic data annotations and supervised learning approaches,” *Nat. Lang. Eng.*, vol. 18, no. 1, pp. 109–145, 2012, doi: 10.1017/S1351324911000167.

[44] N. El-Fishawy, A. Hamouda, G. M. Attiya, and M. Atef, “Arabic summarization in Tw itter social network,” *Ain Shams Eng. J.*, vol. 5, no. 2, pp. 411–420, 2014, doi: 10.1016/j.asej.2013.11.002.

[45] L. S. Mohammed Salem Binwahlan, N. Salim, “Swarm based features selection for text summarization,” *Int. J. Comput. Sci. Netw. Secur.*, vol. 9, no. 1, pp. 175–179.

[46] M. S. Binwahlan, N. Salim, and L. Suanmali, “Swarm Based Text Summarization,” *2009 Int. Assoc. Comput. Sci. Inf. Technol. - Spring Conf. IACSIT-SC 2009*, pp. 145–150, 2009, doi: 10.1109/IACSIT-SC.2009.61.

[47] M. S. Binwahlan, N. Salim, and L. Suanmali, “Swarm diversity based text summarization,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5864 LNCS, no. PART 2, pp. 216–225, 2009, doi: 10.1007/978-3-642-10684-2_24.

[48] L. S. Mohammed Salem Binwahlan , Naomie Salim, “Fuzzy Swarm Based Text Summarization,” *J. Comput. Sci.*, vol. 5, no. 5, p. :338-346, 2009.

[49] M. Franco-Salvador, P. Rosso, and M. Montes-y-Gómez, “A systematic study of knowledge graph analysis for cross-language plagiarism detection,” *Inf. Process. Manag.*, vol. 52, no. 4, pp. 550–570, Jul. 2016, doi: 10.1016/j.ipm.2015.12.004.

[50] R. Ferreira *et al.*, “Assessing sentence scoring techniques for extractive text summarization,” *Expert Syst. Appl.*, vol. 40, no. 14, pp. 5755–5764, Oct. 2013, doi: 10.1016/j.eswa.2013.04.023.

[51] G. A. Miller, “WordNet,” *Commun. ACM*, vol. 38, no. 11, pp. 39–41, Nov. 1995, doi: 10.1145/219717.219748.

[52] P. Oram, “WordNet: An electronic lexical database. Christiane Fellbaum (Ed.). Cambridge, MA: MIT Press, 1998. Pp. 423.,” *Appl. Psycholinguist.*, vol. 22, no. 1, pp. 131–134, Mar. 2001, doi: 10.1017/S0142716401221079.

[53] A. Nenkova and K. McKeown, “A Survey of Text Summarization

Techniques," in *Mining Text Data*, Boston, MA: Springer US, 2012, pp. 43–76.

[54] G. Erkan and D. R. Radev, "LexRank: Graph-based Lexical Centrality as Salience in Text Summarization," *J. Artif. Intell. Res.*, vol. 22, pp. 457–479, Dec. 2004, doi: 10.1613/jair.1523.

[55] P. Mehta and P. Majumder, "Effective aggregation of various summarization techniques," *Inf. Process. Manag.*, vol. 54, no. 2, pp. 145–158, Mar. 2018, doi: 10.1016/j.ipm.2017.11.002.

[56] and P. T. Mihalcea, Rada, "Textrank: Bringing order into text," *Proc. 2004 Conf. Empir. methods Nat. Lang. Process.*, pp. 404–411, 2004.

[57] S. Wang, X. Zhao, B. Li, B. Ge, and D. Tang, "Integrating Extractive and Abstractive Models for Long Text Summarization," in *2017 IEEE International Congress on Big Data (BigData Congress)*, Jun. 2017, pp. 305–312, doi: 10.1109/BigDataCongress.2017.46.

[58] K. Al-Sabahi, Z. Zhang, J. Long, and K. Alwesabi, "An Enhanced Latent Semantic Analysis Approach for Arabic Document Summarization," *Arab. J. Sci. Eng.*, vol. 43, no. 12, pp. 8079–8094, Dec. 2018, doi: 10.1007/s13369-018-3286-z.

[59] M. Mohamed and M. Oussalah, "SRL-ESA-TextSum: A text summarization approach based on semantic role labeling and explicit semantic analysis," *Inf. Process. Manag.*, vol. 56, no. 4, pp. 1356–1372, Jul. 2019, doi: 10.1016/j.ipm.2019.04.003.

[60] and T. Y. Kobayashi, Hayato, Masaki Noguchi, "Summarization based on embedding distributions.," *Proc. 2015 Conf. Empir. methods Nat. Lang. Process.*, 2015.

[61] and M. L. N. Chen, Laifu, "Sentence selective neural extractive summarization with reinforcement learning," *2019 11th Int. Conf. Knowl. Syst. Eng.*, 2019.

[62] J. M. Sanchez-Gomez, M. A. Vega-Rodríguez, and C. J. Pérez, "Experimental analysis of multiple criteria for extractive multi-document text summarization," *Expert Syst. Appl.*, vol. 140, p. 112904, Feb. 2020, doi: 10.1016/j.eswa.2019.112904.

[63] Y. K. Meena and D. Gopalani, "Evolutionary Algorithms for Extractive Automatic Text Summarization," *Procedia Comput. Sci.*, vol. 48, pp. 244–249, 2015, doi: 10.1016/j.procs.2015.04.177.

[64] A. D. I. T. I. Kumar, A. K. S. H. I., & Sharma, "Systematic literature review of fuzzy logic based text summarization," *Iran. J. Fuzzy Syst.*, vol. 16, pp. 45–59, 2019, doi: 10.22111/ijfs.2019.4906.

[65] M. A. Nazari, N., & Mahdavi, "A survey on automatic text summarization," *J. AI Data Min.*, vol. 7, no. 1, pp. 121–135, 2019, doi: 10.22044/jadm.2018.6139.1726.

[66] S. Gupta and S. K. Gupta, "Abstractive summarization: An overview of the state of the art," *Expert Syst. Appl.*, vol. 121, pp. 49–65, May 2019, doi: 10.1016/j.eswa.2018.12.011.

[67] J. Ganesan, Kavita; Zhai, ChengXiang; Han, "Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions," *Pap. Present. Proc. 23rd Int. Conf. Comput. Linguist. Beijing, China.*, 2010.

[68] and A. S. Gupta, Vanyaa, Neha Bansal, "Text summarization for big data: A comprehensive survey," *Pap. Present. Int. Conf. Innov. Comput. Commun. Singapore.*, pp. 503–516, 2019.

[69] V. V. Litton J Kurisinkel, Yue Zhang, "Abstractive Multi-document Summarization by Partial Tree Extraction, Recombination and Linearization," *Pap. Present. Proc. Eighth Int. Jt. Conf. Nat. Lang. Process.*, vol. Volume 1:, pp. 812–821, 2017.

[70] G. L. Pierre-Etienne Genest, "Fully Abstractive Approach to Guided Summarization," *Pap. Present. Proc. 50th Annu. Meet. Assoc. Comput. Linguist. Short Pap.*, vol. 2, pp. 354–358, 2012.

[71] N. Moratanch and S. Chitrakala, "A survey on abstractive text summarization," in *2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT)*, Mar. 2016, pp. 1–7, doi: 10.1109/ICCPCT.2016.7530193.

[72] N. Okumura and T. Miura, "Automatic labelling of documents based on ontology," in *2015 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, Aug. 2015, pp. 34–39, doi: 10.1109/PACRIM.2015.7334805.

[73] A. Khan, N. Salim, and Y. Jaya Kumar, "A framework for multi-document abstractive summarization based on semantic role labelling," *Appl. Soft Comput.*, vol. 30, pp. 737–747, May 2015, doi: 10.1016/j.asoc.2015.01.070.

[74] L. Hou1, P. Hu1, and and Chao Bei, "Abstractive Document Summarization via Neural Model with Joint Attention," *Natl. CCF Conf. Nat. Lang. Process. Chinese Comput.*, pp. 329–338, 2017.

[75] and P. S. Rehurek, Radim, "Software framework for topic modelling with large corpora," *Proc. Lr. 2010 Work. new challenges NLP Fram.*, 2010.

[76] and A. S. Kouris, Panagiotis, Georgios Alexandridis, "Abstractive Text Summarization Based on Deep Learning and Semantic Content Generalization," *Pap. Present. Proc. 57th Annu. Meet. Assoc. Comput. Linguist. Florence, Italy*, 2019.

[77] S. G. Soumi Dutta, Vibhash Chandra, Kanav Mehra and A. K. D. and S. Ghosh, "Summarizing Microblogs During Emergency Events: A Comparison of Extractive Summarization Algorithms," *Pap. Present. Int. Conf. Emerg. Technol. Data Min. Inf. Secur. (IEMIS 2018), Kolkata, India*, 2019.

[78] Mingzhen Chen and Yu Song, "Summarization of text clustering based vector space model," in *2009 IEEE 10th International Conference on Computer-Aided Industrial Design & Conceptual Design*, Nov. 2009, pp. 2362–2365, doi: 10.1109/CAIDCD.2009.5375265.

[79] M. Melucci, "Vector-Space Model," in *Encyclopedia of Database Systems*, Boston, MA: Springer US, 2009, pp. 3259–3263.

[80] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Process. Manag.*, vol. 24, no. 5, pp. 513–523, Jan. 1988, doi: 10.1016/0306-4573(88)90021-0.

[81] T. Chen, R. Xu, Y. He, and X. Wang, "Improving sentiment analysis via sentence type classification using BiLSTM-CRF and CNN," *Expert Syst. Appl.*, vol. 72, pp. 221–230, Apr. 2017, doi: 10.1016/j.eswa.2016.10.065.

[82] R. Narayanan, B. Liu, and A. Choudhary, "Sentiment analysis of conditional sentences," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing Volume 1 - EMNLP '09*, 2009, vol. 1, p. 180, doi: 10.3115/1699510.1699534.

[83] R. Xia, F. Xu, J. Yu, Y. Qi, and E. Cambria, "Polarity shift detection, elimination and ensemble: A three-stage model for document-level sentiment analysis," *Inf. Process. Manag.*, vol. 52, no. 1, pp. 36–45, Jan. 2016, doi: 10.1016/j.ipm.2015.04.003.

[84] U. Farooq, "Negation Handling in Sentiment Analysis at Sentence Level," *J. Comput.*, pp. 470–478, 2017, doi: 10.17706/jcp.12.5.470-478.

[85] B. Liu, "Sentiment Analysis and Opinion Mining," *Synth. Lect.*

Hum. Lang. Technol., vol. 5, no. 1, pp. 1–167, May 2012, doi: 10.2200/S00416ED1V01Y201204HLT016.

[86] *Combinatorial Optimization*, vol. 21. Berlin/Heidelberg: Springer-Verlag, 2006.

[87] B. Carterette, "An analysis of NP-completeness in novelty and diversity ranking," *Inf. Retr. Boston.*, vol. 14, no. 1, pp. 89–106, Feb. 2011, doi: 10.1007/s10791-010-9157-1.

[88] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence*. Oxford University Press, 1999.

[89] S. Goss, R. Beckers, J. L. Deneubourg, S. Aron, and J. M. Pasteels, "HOW TRAIL LAYING AND TRAIL FOLLOWING CAN SOLVE FORAGING PROBLEMS FOR ANT COLONIES Unit of Behavioural Ecology , CP 231 Universite Libre de Bruxelles 1050 Bruxelles Belgium Introduction One of the most striking features of an ant colony â€™ s behaviour is it," 1990.

[90] J.-L. Deneubourg, S. Aron, S. Goss, and J. M. Pasteels, "The self-organizing exploratory pattern of the argentine ant," *J. Insect Behav.*, vol. 3, no. 2, pp. 159–168, Mar. 1990, doi: 10.1007/BF01417909.

[91] C. Blum, "Ant colony optimization: Introduction and recent trends," *Phys. Life Rev.*, vol. 2, no. 4, pp. 353–373, Dec. 2005, doi: 10.1016/j.plrev.2005.10.001.

[92] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Trans. Syst. Man, Cybern. Part B*, vol. 26, no. 1, pp. 29–41, Feb. 1996, doi: 10.1109/3477.484436.

[93] J. K. Lenstra, D. B. Shmoys, and É. Tardos, "Approximation algorithms for scheduling unrelated parallel machines," *Math. Program.*, vol. 46, no. 1–3, pp. 259–271, Jan. 1990, doi: 10.1007/BF01585745.

[94] D. Merkle and M. Middendorf, "Ant Colony Optimization," *Eur. J. Oper. Res.*, vol. 168, no. 1, pp. 269–271, Jan. 2006, doi: 10.1016/j.ejor.2004.09.013.

[95] J. G. Falcón-Cardona, G. Leguizamón, C. A. Coello Coello, and M. G. Castillo Tapia, "Multi-objective Ant Colony Optimization: An Updated Review of Approaches and Applications," *Intell. Syst. Ref. Libr.*, vol. 218, no. April, pp. 1–32, 2022, doi: 10.1007/978-981-16-8930-7_1.

[96] Marco Dorigo; Thomas Stützle, “Ant Colony Optimization Algorithms for the Traveling Salesman Problem,” 2004, p. pp.65-119.

[97] T. Oscar Cordon, farncisco Herrera, “1-Cordon-Herrera-Stuetzle,” 2002.

[98] M. López-ibáñez, T. Stützle, and M. Dorigo, *Ant Colony Optimization : A Component-Wise Overview*. 2018.

[99] M. S. R. Monteiro, D. B. M. M. Fontes, and F. A. C. C. Fontes, “Ant Colony Optimization: a literature survey,” *FEP Work. Pap.*, no. December, 2012, [Online]. Available: http://ideas.repec.org/p/por/fepwps/474.html.

[100] V. Maniezzo, L. M. Gambardella, and F. De Luigi, “5 . Ant Colony Optimization,” pp. 1–21, 1991.

[101] J. Ning, C. Zhang, P. Sun, and Y. Feng, “Comparative study of ant colony algorithms for multi-objective optimization,” *Inf.*, vol. 10, no. 1, pp. 1–19, 2018, doi: 10.3390/info10010011.

[102] M. Dorigo and T. Stützle, “Ant Colony Optimization: Overview and Recent Advances,” 2019, pp. 311–351.

[103] E. Lloret, L. Plaza, and A. Aker, “The challenging task of summary evaluation: an overview,” *Lang. Resour. Eval.*, vol. 52, no. 1, pp. 101–148, Mar. 2018, doi: 10.1007/s10579-017-9399-2.

[104] C. Y. Lin, “Rouge: A package for automatic evaluation of summaries,” *Proc. Work. text Summ. branches out (WAS 2004)*, no. 1, pp. 25–26, 2004, [Online]. Available: papers2://publication/uuid/5DDA0BB8-E59F-44C1-88E6-2AD316DAEF85.

[105] Z. Ali and Sb. Bhaskar, “Basic statistical tools in research and data analysis,” *Indian J. Anaesth.*, vol. 60, no. 9, p. 662, 2016, doi: 10.4103/0019-5049.190623.

[106] O. Kolchyna, T. T. P. Souza, P. Treleaven, and T. Aste, “Twitter Sentiment Analysis: Lexicon Method, Machine Learning Method and Their Combination,” 2015, [Online]. Available: http://arxiv.org/abs/1507.00955.

[107] Y. Xie, F. Sun, Y. Deng, Y. Li, and B. Ding, “Factual Consistency Evaluation for Text Summarization via Counterfactual Estimation,” *Find. Assoc. Comput. Linguist. Find. ACL EMNLP 2021*, pp. 100–110, 2021, doi: 10.18653/v1/2021.findings-emnlp.10.

[108] P. Willett, “The Porter stemming algorithm: then and now,” *Program*, vol. 40, no. 3, pp. 219–223, Jul. 2006, doi: 10.1108/00330330610681295.

[109] E. Lloret, E. Boldrini, T. Vodolazova, P. Martínez-Barco, R. Muñoz, and M. Palomar, “A novel concept-level approach for ultra-concise opinion summarization,” *Expert Syst. Appl.*, vol. 42, no. 20, pp. 7148–7156, 2015, doi: 10.1016/j.eswa.2015.05.026.

[110] R. Alqaisi, W. Ghanem, and A. Qaroush, “Extractive Multi-Document Arabic Text Summarization Using Evolutionary Multi-Objective Optimization With K-Medoid Clustering,” *IEEE Access*, vol. 8, pp. 228206–228224, 2020, doi: 10.1109/ACCESS.2020.3046494.

الخلاصـة

في التلخيص التلقائي للنص، تُستخدم خوارزميات الكمبيوتر لإنتاج ملخص مفيد وغني بالمعلومات تلقائيًا لنص الإدخال. لتحليل الرأي العام على الفور حول أي حدث، يتطلب تحليل المشاعر والتنقيب عن الرأي تقنيات تلخيص. الأساليب الأكثر استخدامًا لحل هذه المشكلة هي استخدام طرق التحسين المختلفة، والتي تهتم بإيجاد أفضل حل في ظل قيود معينة. ستستخدم خوارزميات التحسين هذه جودة التلخيص كوظيفة موضوعية. ومع ذلك، في هذا النهج، يتم استخدام خوارزمية أمثليه مستعمرة النمل الهدف الموجهة للمشاعر (SO.ACO)بطريقة فريدة. بدلاً من تغذية المستندات مباشرة إلى الخوارزمية للعثور على أفضل تلخيص، اعتبر هذا النهج المقترح درجة المشاعر وتشابه TF-IDF كمصفوفة المسافة لتحسين مستعمرة النمل. مع الجمل كعقد لمسألة تحسين مستعمرة النمل، والحواف هي المسافة بين كل جملة. تعتبر الجملة الأكثر ارتباطًا بالجمل الأخرى (الجملة الموجودة في منتصف المستند) أهم جملة. وبدلاً من الاعتماد فقط على المسافة كوظيفة موضوعية لخوارزميةAnt Colony Optimization ، يتم النظر في ثلاثة معايير لهذا الهدف: ملاءمة المشاعر وتغطية المحتوى وتقليل التكرار. يتم أخذها في الاعتبار عند البحث عن أفضل حل. أيضًا، تم استخدامه لترقية الوظيفة الاستكشافية التي تعد القاعدة الثانية لبناء الحل فيACO .

مجموعة البيانات المستخدمة في هذا العمل هي DUC 2002 .وهي عبارة عن ٥٠٩ وثيقة. مقسمة إلى ٥٩ موضوعًا. كلهم نصوص ذات صلة بالأخبار. ويتم إجراء التلخيص المدعوم يدويًا بواسطة خبراء.

المقاييس المستخدمة في تقييم النهج المقترح هي Rouge-L و Rouge-2و Rouge-l .علاوة على ذلك، يتم حساب ملاءمة المشاعر وتغطية المحتوى وتقليل التكرار بين الملخصات التي تم إنشاؤها والملخصات اليدوية. ايضا، يتم حساب ارتباط بيرسون بين الخوارزمية التي تم إنشاؤها والتلخيص الذي تم إنشاؤه يدويًا. النتائج التي تم الحصول عليها كالتالي :0.601 Rouge-1، 0.5294:Rouge-2 ، 0.5908 :Rouge-L. وملاءمة المشاعر:1.868 ، تغطية المحتوى: 0.78 ، والتخفيضات في التكرار:0.627 . ومعامل ارتباط بيرسون يساوي 0.8603. هذه النتائج أعلى من نتائج الأساليب الموجودة في المقالات المنشورة، حيث متوسط النسبة المئوية للتحسين على النحو التالي Rouge-1 : 20.23 + ٪ و Rouge-2: + 207.21٪.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعـــة بـابـــل
كلية تكنولوجيا المعلومات
قسم البرمجيات

تلخيص النص الموجه نحو المشاعر باستخدام مستعمرة النمل المحسنة ذات الأهداف الموحدة

رسالة مقدمة إلى
مجلس كلية تكنولوجيا المعلومات -جامعة بابل كجزء من متطلبات
نيل درجة الماجستير في تكنولوجيا المعلومات / البرمجيات

من قبل
عبير رعد حسن علي

بإشـــراف

أ.م.د.رافد صكبان عبود كريم

١٤٤٤ هـ ٢٠٢٢ م