Text Summarization based on Fuzzy Logic

A report submitted in partial fulfillment of the requirements for Summer Project (BCCS-2999)

Bachelor of Technology
in
Computer Science and Engineering

by

Gagandeep Singh	**2021BCS-26**
Bhavana Mekala	**2021BCS-41**
Nelluri Pavithra Sai Lakshmi	**2021BCS-49**

विश्वजीवनामृतं ज्ञानम्

ABV INDIAN INSTITUTE OF INFORMATION TECHNOLOGY AND MANAGEMENT
GWALIOR - 474015

July 2023

CANDIDATES DECLARATION

We hereby certify that the work, which is being presented in the report, entitled **Text Summarization**, in partial fulfillment of the requirement for summer project (BCCS-2999) for **Bachelor of Technology in Computer Science and Engineering** and submitted to the institution is an authentic record of our own work carried out during the period *May 2023* to *July 2023* under the supervision of **Dr. Santosh Singh Rathore**. We also cited the reference about the text(s) from where they have been taken.

Date:	Signature of the Candidate
Date:	Signature of the Candidate
Date:	Signature of the Candidate

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Date:	Signature of the Supervisor(s)

ABSTRACT

Due to the vast and ever-growing amount of textual content available online, as well as in archives of news articles, scientific reports, legal documents, and more, Automatic Text Summarization (ATS) is gaining increasing importance. Various ATS techniques have been divided into various categories but broadly These techniques can be categorized as extractive, or abstractive. While numerous studies have been conducted on acquiring datasets, methods, and techniques, there is a scarcity of comprehensive papers providing a detailed overview of text summarization based on fuzzy logic using fuzzy rules. This paper aims to address this gap by offering a systematic and extensive approach to do the same. Additionally, the report describes some basic approaches like tf-idf and other methods which use fuzzy logic but does not involve fuzzy rules. Finally, the report concludes with recommendations highlighting techniques for more optimization and challenges that researchers may encounter in their text summarization(using fuzzy logic) endeavors.

Keywords: Automatic Text Summarization (ATS), Fuzzy logics.

ACKNOWLEDGEMENTS

We are highly indebted to **Dr. Santosh Singh Rathore**, and are obliged for giving us the autonomy of functioning and experimenting with ideas. We would like to take this opportunity to express our profound gratitude to him not only for his academic guidance but also for their personal interest in our project and constant support coupled with confidence boosting and motivating sessions which proved very fruitful and were instrumental in infusing self-assurance and trust within us. The nurturing and blossoming of the present work is mainly due to their valuable guidance, suggestions, astute judgment, constructive criticism and an eye for perfection. Our mentor always answered myriad of our doubts with smiling graciousness and prodigious patience, never letting us feel that we are novices by always lending an ear to our views, appreciating and improving them and by giving us a free hand in our project. It's only because of his overwhelming interest and helpful attitude, the present work has attained the stage it has.

Finally, we are grateful to our Institution and colleagues whose constant encouragement served to renew our spirit, refocus our attention and energy and helped us in carrying out this work.

Gagandeep Singh
Bhavana Mekala
Nelluri Pavithra Sai Lakshmi

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Problem Formulation

Manual text summarization is a time-consuming and costly process, and it becomes impractical when dealing with such massive amounts of text. With the proliferation of the internet and the availability of vast amounts of information, individuals often find themselves overwhelmed by the sheer volume of documents to process. This has led researchers to explore technological solutions for automatic text summarization.

1.2 Background

In the age of the information explosion, the availability of large amounts of textual data has made it difficult to process and understand this information efficiently. Text summarization has emerged as a useful solution for dealing with large amounts of text, providing users with a concise and consistent summary that captures the essence of the original content. This task is particularly relevant in scenarios where users need to quickly grasp key points in large documents, articles, news, or research papers. Automatic text summarization aims to generate concise summaries that capture the key sentences and relevant information from the original text ,objective is to deliver information quickly without losing the essence of the original document.two most common summarization techniques are extractive and abstractive summarization. Extractive summarization extracts sentences directly from the source text retaining the wording and structure of the original sentences. It relies on indicative keywords to identify important sentences while Abstractive summarization generates new sentences by rewriting and paraphrasing the content. It uses informative keywords to capture the main themes of the text. Text summarization has many applications. A news summary service can provide readers with a quick overview of current events. In the legal field, on the other hand, summaries help legal professionals efficiently process large volumes of litigation documents. Researchers can use this overview to quickly sift through large numbers of research papers. Additionally, chatbots and virtual assistants often use summaries to provide concise answers to user queries.

1.3 Objective

Our primary objective is to explore both abstractive and extractive text summarization techniques, understand their unique challenges, and evaluate their performance to determine their practicality in real-world applications.

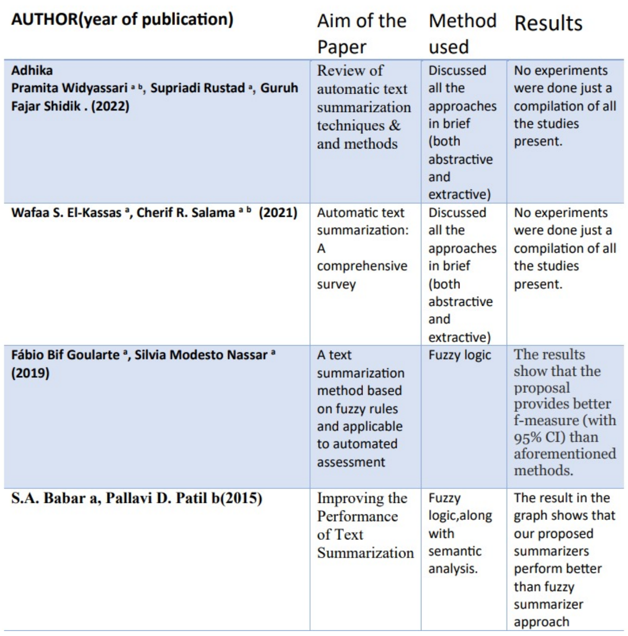
1. Build Extractive Text Summarization using Fuzzy Inference System:

Here our objective is to develop an extractive text summarization approach based on a fuzzy inference system. Extractive summarization involves selecting and extracting important sentences or phrases from the original text to form a summary. Fuzzy logic provides a way to handle uncertainty and vagueness in data, making it suitable for modeling the linguistic rules involved in the sentence extraction process. By building an extractive summarization system with a fuzzy inference system, we aim to identify and extract salient information from the input text based on defined linguistic rules.

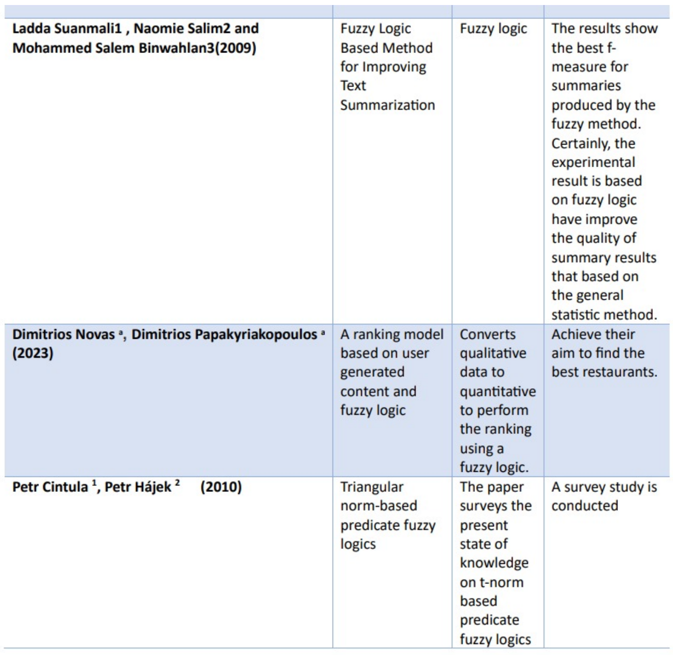
2.Implement Abstractive Text Summarization with BART:

Abstractive summarization aims to generate concise and coherent summaries in a more human-like fashion by paraphrasing and rephrasing the original content. The BART model, being a transformer-based architecture, has shown promising results in various natural language processing tasks, including text summarization. By utilizing BART, we aim to explore its capabilities in generating high-quality abstractive summaries from input text documents.

1.4 Literature Review

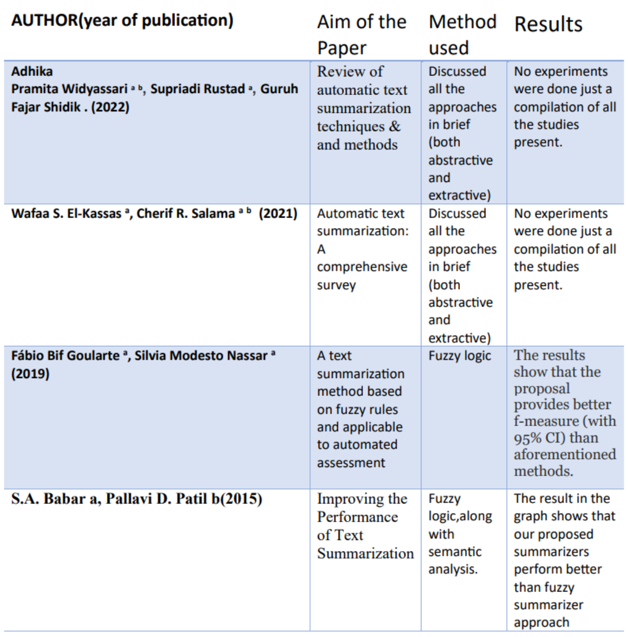


AUTHOR(year of publication)	Aim of the Paper	Method used	Results
Adhika Pramita Widyassari [a b], Supriadi Rustad [a], Guruh Fajar Shidik . (2022)	Review of automatic text summarization techniques & and methods	Discussed all the approaches in brief (both abstractive and extractive)	No experiments were done just a compilation of all the studies present.
Wafaa S. El-Kassas [a], Cherif R. Salama [a b] (2021)	Automatic text summarization: A comprehensive survey	Discussed all the approaches in brief (both abstractive and extractive)	No experiments were done just a compilation of all the studies present.
Fábio Bif Goularte [a], Silvia Modesto Nassar [a] (2019)	A text summarization method based on fuzzy rules and applicable to automated assessment	Fuzzy logic	The results show that the proposal provides better f-measure (with 95% CI) than aforementioned methods.
S.A. Babar a, Pallavi D. Patil b(2015)	Improving the Performance of Text Summarization	Fuzzy logic,along with semantic analysis.	The result in the graph shows that our proposed summarizers perform better than fuzzy summarizer approach

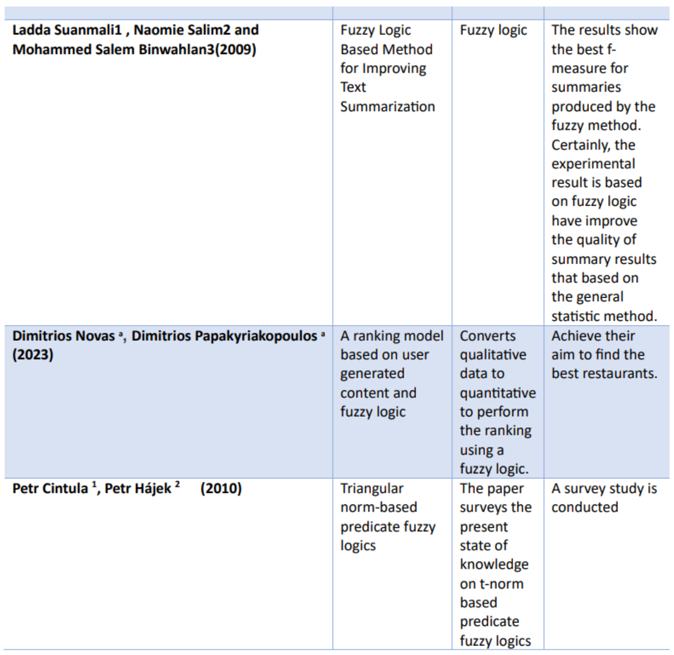


Ladda Suanmali1 , Naomie Salim2 and Mohammed Salem Binwahlan3(2009)	Fuzzy Logic Based Method for Improving Text Summarization	Fuzzy logic	The results show the best f-measure for summaries produced by the fuzzy method. Certainly, the experimental result is based on fuzzy logic have improve the quality of summary results that based on the general statistic method.
Dimitrios Novas [a], Dimitrios Papakyriakopoulos [a] (2023)	A ranking model based on user generated content and fuzzy logic	Converts qualitative data to quantitative to perform the ranking using a fuzzy logic.	Achieve their aim to find the best restaurants.
Petr Cintula [1], Petr Hájek [2] (2010)	Triangular norm-based predicate fuzzy logics	The paper surveys the present state of knowledge on t-norm based predicate fuzzy logics	A survey study is conducted

graphicx



AUTHOR(year of publication)	Aim of the Paper	Method used	Results
Adhika Pramita Widyassari [a, b], Supriadi Rustad [a], Guruh Fajar Shidik . (2022)	Review of automatic text summarization techniques & and methods	Discussed all the approaches in brief (both abstractive and extractive)	No experiments were done just a compilation of all the studies present.
Wafaa S. El-Kassas [a], Cherif R. Salama [a b] (2021)	Automatic text summarization: A comprehensive survey	Discussed all the approaches in brief (both abstractive and extractive)	No experiments were done just a compilation of all the studies present.
Fábio Bif Goularte [a], Silvia Modesto Nassar [a] (2019)	A text summarization method based on fuzzy rules and applicable to automated assessment	Fuzzy logic	The results show that the proposal provides better f-measure (with 95% CI) than aforementioned methods.
S.A. Babar a, Pallavi D. Patil b(2015)	Improving the Performance of Text Summarization	Fuzzy logic,along with semantic analysis.	The result in the graph shows that our proposed summarizers perform better than fuzzy summarizer approach



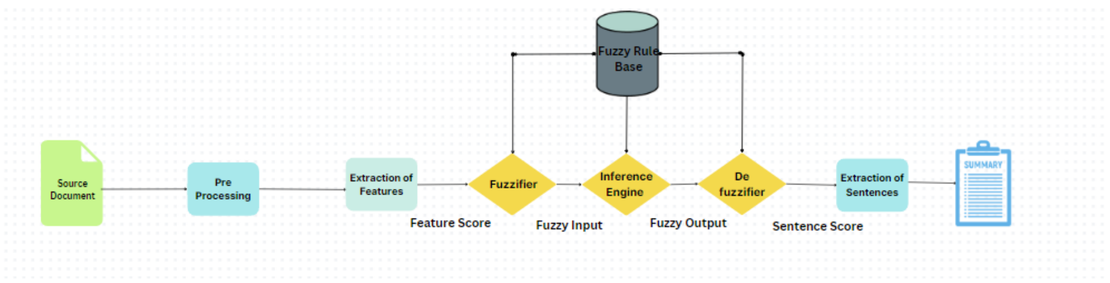
Ladda Suanmali1 , Naomie Salim2 and Mohammed Salem Binwahlan3(2009)	Fuzzy Logic Based Method for Improving Text Summarization	Fuzzy logic	The results show the best f-measure for summaries produced by the fuzzy method. Certainly, the experimental result is based on fuzzy logic have improve the quality of summary results that based on the general statistic method.
Dimitrios Novas [a], Dimitrios Papakyriakopoulos [a] (2023)	A ranking model based on user generated content and fuzzy logic	Converts qualitative data to quantitative to perform the ranking using a fuzzy logic.	Achieve their aim to find the best restaurants.
Petr Cintula [1], Petr Hájek [2] (2010)	Triangular norm-based predicate fuzzy logics	The paper surveys the present state of knowledge on t-norm based predicate fuzzy logics	A survey study is conducted

CHAPTER 2

IMPLEMENTATION/EXECUTION OF PROJECT

2.1 Extractive Summarization

2.1.1 Text Summarization using Fuzzy rules



Generally, text summarization using a fuzzy inference system involves the following necessary steps-

Preprocessing:

1. Text Cleaning: · Remove special characters, symbols, and punctuation marks. · Convert the text to lowercase for case insensitivity. · Remove extra spaces and tabs.

2. Tokenization: · Break the text into individual words or tokens. This step is essential for further processing and analysis.

3. Stopword Removal: · Eliminate common and non-informative words such as "the," "and," "in," "is," etc., which do not contribute significantly to the summary.

4. Lemmatization or Stemming: · Reduce words to their base or root form. This helps in consolidating similar words, e.g., "running" and "ran" to "run."

5. Part-of-Speech Tagging (POS Tagging) : Assign grammatical tags to each word,

such as noun, verb, adjective, etc. This information can be useful for certain summarization approaches, but it is not always required.we will use this while extracting our thematic features from the text.

6.Sentence Segmentation: · Split the text into individual sentences. A summary typically consists of a selection of important sentences.

Text Representation:

User inputs:

Title:

Note that the user needs to enter the title of the text manually on the website. this entered title will help in calculating our title feature score and thematic words feature. Though in our code we have calculated the thematic word feature score using text which will definitely increase the time of calculation in extracting the words and their matching with each sentence while finding similarities it will give us more accuracy than that method of calculating thematic word feature score using title only.

Number of lines of summary:

User also needs to enter the number of lines of summary he/she needs so that once we have ordered the sentences on the basis of their sentence scores we can extract that number of top sentences to form the summary.

Text:

User can enter the text of which summary he/she wants. this text can be different pages or paragraphs. but in order to proceed we will convert the whole into a single paragraph

Linguistic Variable Definition:

a. Determine linguistic variables related to sentence importance, such as "very important," "important," "somewhat important," etc.

b. Define membership functions for each linguistic variable/features, specifying their shapes and ranges.

Defining text features: After completing the preprocessing steps, every sentence in the document is converted into an attribute vector that contains various features. These features serve as attributes aiming to represent the information relevant to their specific purpose. Our attention is directed towards eight distinct features for each sentence. Each of these features is assigned a numerical value ranging from ’0’ to ’1’. The eight features are as follows -:

1. Title Feature Score: The similarity between a sentence and the document's title is calculated by this characteristic. It counts the number of times each word from the phrase appears in the title and gives a score based on how many times.

Methodology: The statement and the title should be tokenized into words in order to implement this feature. To determine the score, divide the total number of words in the sentence that appear in title by the number of words that appear in the title.

F1(S) = No.Title word in S / No.Word in Title

2. Sentence Length: Using the amount of words in each sentence, this feature determines how long each sentence is. Sentences that are longer are often thought to carry more information than those that are shorter might.Short sentences like datelines and author names, which usually appear in news items, can be filtered away with this function. It is not anticipated that the short sentences will be part of the summary.

Methodology:

The procedure that follows counts the number of words in each sentence after tokenizing it into words.it is the ratio of the number of words occurring in the sentence over the number of words occurring in the longest sentence of the document.

F2(S) = No.Word occurring in S / No.Word occurring in longest sentence

3. Sentence Position the importance of the sentences is shown by their placement in the text. This feature can include a number of different things, such as where a sentence is located inside a section, paragraph, etc. It is suggested that the first sentence has the most priority.

Methodology: Suppose the paragraph's first five sentences are taken into account for calculating this feature's score.

F3(S) = 5/5 for 1st, 4/5 for 2nd, 3/5 for 3rd, 2/5 for 4th, 1/5 for 5th, 0/5 for other sentences

4. Sentence Similarity The cosine similarity metric is used to determine how similar each sentence is to each other for each sentence S, yielding a number between 0 and 1. The vectors Si and Sj are used to indicate the term weights wi and wj of terms t to n.

Methodology:

The ratio of a sentence's summary of sentence similarity to all other sentences over the maximum of summary is used to calculate a sentence's score for this attribute.

5. Numerical score ratio:

This determines the proportion of numerical data to all the words in a sentence. It aids in identifying sentences that contain statistics or numerical data. The numerical score ratio is calculated by counting the number of numerical data (such as numbers) in a sentence and dividing it by the total number of words in the sentence.

F5(S) = No. Numerical data in S / Sentence Length (S)

6. Thematic word feature score: This feature calculates the score by counting the number of thematic words (such as nouns) contained in a sentence. The frequency of thematic words per sentence is an important characteristic because words that occur frequently within a document are likely related to the topic. The number of thematic words indicates the terms with the highest degree of relevance. The steps involved in calculating the thematic word score are as follows. Tokenize the sentence, remove stop words, and tag the parts of speech. Identify the thematic words in the sentence. Calculate the score as the ratio of the number of thematic words that occur in the sentence to the maximum number of thematic words in the sentence.see thematic words can be extracted from the text as well as title of the text .
F6(S) = No. Thematic word in S/Max No. Thematic word

F6(S) = No. Thematic word in S/Max(No. Thematic word)

7. Sentence Weight: as we know TF-IDF is one of the oldest methods of computing sentence scores so we are calculating the sentence weights which will be same as sentence scores returned by it.

Methodology:

TF-IDF Model:

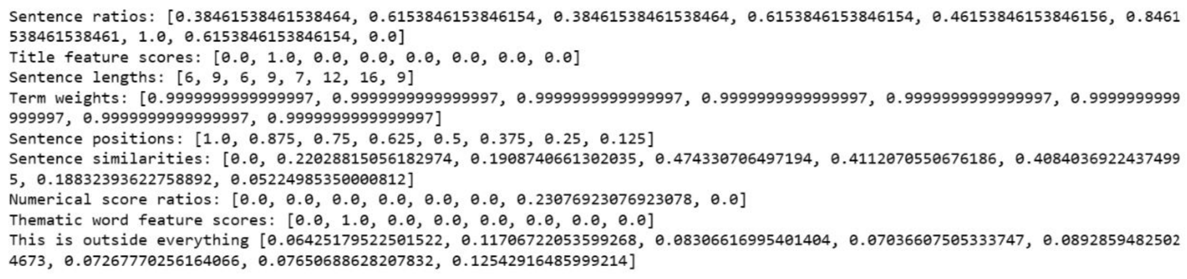
First step is Preprocessing of text followed by creating the Frequency matrix of the words in each sentence.Next, we need to calculate the frequency of words in each sentence. Here, each sentence serves as the key, and a dictionary of word frequency serves as the value. Calculating the Term Frequency and generating a Term Frequency Matrix: Here we need to determine the term frequency of each word. Term frequency (TF) means how often a term occurs in a document.

Term Frequency(t) = Number of times term t appears in a document / number of terms in the document in total Words with the same frequency have similar TF scores.

Creating a table for documents per words Next We need to determine how many sentences the word is present by using a matrix of documents per word. Calculate IDF and generate a matrix.Next we need to calculate Inverse document frequency matrix. A weight called Inverse Document Frequency (IDF) indicates how frequently a word is used. The lower its score, the more frequently it is used in documents.

DF(t) = loge(Total number of documents / Number of documents with term t in it)

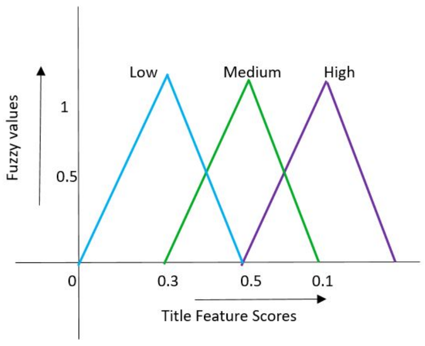
While calculating TF-IDF and generating a matrix we will be generating a new matrix by multiplying the values from both the previously generated matrices. We are using the Tf-IDF score of words in a sentence to give weight to the sentence. So this final values for each sentence will act as a sentence weights for each sentence.



```
Sentence ratios: [0.38461538461538464, 0.6153846153846154, 0.38461538461538464, 0.6153846153846154, 0.46153846153846156, 0.8461
538461538461, 1.0, 0.6153846153846154, 0.0]
Title feature scores: [0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
Sentence lengths: [6, 9, 6, 9, 7, 12, 16, 9]
Term weights: [0.9999999999999997, 0.9999999999999997, 0.9999999999999997, 0.9999999999999997, 0.9999999999999997, 0.9999999999
999997, 0.9999999999999997, 0.9999999999999997]
Sentence positions: [1.0, 0.875, 0.75, 0.625, 0.5, 0.375, 0.25, 0.125]
Sentence similarities: [0.0, 0.22028815056182974, 0.1908740661302035, 0.474330706497194, 0.4112070550676186, 0.4084036922437499
5, 0.18832393622758892, 0.05224985350000812]
Numerical score ratios: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.23076923076923078, 0.0]
Thematic word feature scores: [0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
This is outside everything [0.06425179522501522, 0.11706722053599268, 0.08306616995401404, 0.07036607505333747, 0.0892859482502
4673, 0.07267770256164066, 0.07650688628207832, 0.12542916485999214]
```

Fuzzification converts numerical scores into linguistic terms using a triangular membership function obviously you can use other fuzzy membership functions but just to make our calculations fast and precise comparatively we utilized this. also, you can use different membership functions for different features depending on the domain knowledge but here we will be using just one membership function to evaluate fuzzy values for each feature. Fuzzy rules are then applied to determine the importance level of each sentence. Defuzzification transforms the fuzzy output into a categorical representation to categorize sentences as unimportant, average, or important. Though I am not using it in our project which I will explain further

Fuzzification:



Fuzzification is the process of mapping numerical values to linguistic terms or fuzzy sets. here, the triangular membership function is used for fuzzification. A triangular membership function is a common choice because it allows for a gradual transition between linguistic terms.

Each score, which represents the importance of a sentence, is fuzzified using the triangular membership function. The triangular membership function defines three linguistic terms: LOW, MEDIUM, and HIGH. These terms are represented by triangular-shaped fuzzy sets.See these terms are different for different features though the meaning remains the same for example in the title feature score I have used start middle and end as the linguistic terms which are having the same meanings as of low,middle and high

Let's assume we have a sentence with a sentence length score(its one of our feature) of 0.3. To fuzzify this score, we calculate the degree of membership for each linguistic term using the triangular membership functions.

For the LOW term: Degree of membership = 1 - (0.3 - 0)/(0.4 - 0) = 0.75.

For the MEDIUM term: Degree of membership = (0.3 - 0.2)/(0.4 - 0.2) = 0.5.

For the HIGH term: Degree of membership = (0.3 - 0.2)/(0.4 - 0.2) = 0.5.

So, the fuzzified values for this sentence's length would be:

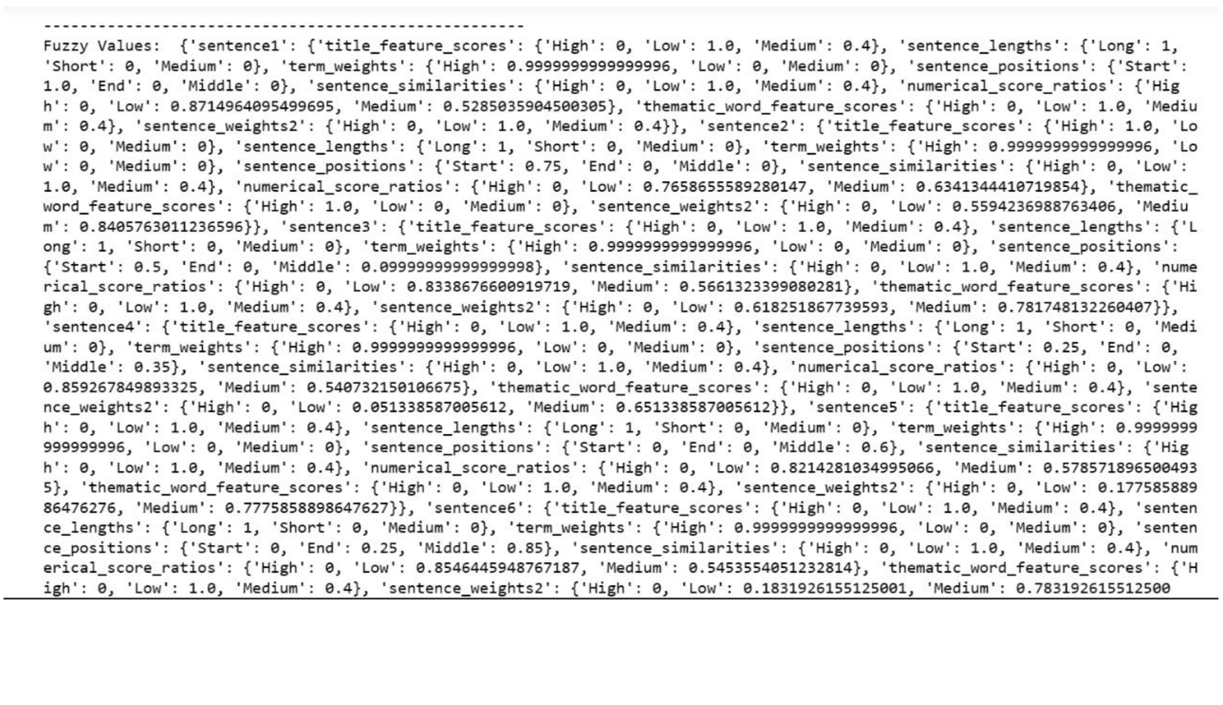
LOW: 0.75

MEDIUM: 0.5

HIGH: 0.5

This process is repeated for all sentences, fuzzifying their length scores into the corresponding linguistic terms. And we will calculate fuzzy values for each feature like this corresponding to each sentence.

In this code, the fuzzy rules and fuzzy values are used together to calculate the


```
----------------------------------------------------
Fuzzy Values:  {'sentence1': {'title_feature_scores': {'High': 0, 'Low': 1.0, 'Medium': 0.4}, 'sentence_lengths': {'Long': 1,
'Short': 0, 'Medium': 0}, 'term_weights': {'High': 0.9999999999999996, 'Low': 0, 'Medium': 0}, 'sentence_positions': {'Start':
1.0, 'End': 0, 'Middle': 0}, 'sentence_similarities': {'High': 0, 'Low': 1.0, 'Medium': 0.4}, 'numerical_score_ratios': {'Hig
h': 0, 'Low': 0.8714964095499695, 'Medium': 0.5285035904500305}, 'thematic_word_feature_scores': {'High': 0, 'Low': 1.0, 'Mediu
m': 0.4}, 'sentence_weights2': {'High': 0, 'Low': 1.0, 'Medium': 0.4}}, 'sentence2': {'title_feature_scores': {'High': 1.0, 'Lo
w': 0, 'Medium': 0}, 'sentence_lengths': {'Long': 1, 'Short': 0, 'Medium': 0}, 'term_weights': {'High': 0.9999999999999996, 'Lo
w': 0, 'Medium': 0}, 'sentence_positions': {'Start': 0.75, 'End': 0, 'Middle': 0}, 'sentence_similarities': {'High': 0, 'Low':
1.0, 'Medium': 0.4}, 'numerical_score_ratios': {'High': 0, 'Low': 0.7658655589280147, 'Medium': 0.6341344410719854}, 'thematic_
word_feature_scores': {'High': 1.0, 'Low': 0, 'Medium': 0}, 'sentence_weights2': {'High': 0, 'Low': 0.5594236988763406, 'Mediu
m': 0.8405763011236596}}, 'sentence3': {'title_feature_scores': {'High': 0, 'Low': 1.0, 'Medium': 0.4}, 'sentence_lengths': {'L
ong': 1, 'Short': 0, 'Medium': 0}, 'term_weights': {'High': 0.9999999999999996, 'Low': 0, 'Medium': 0}, 'sentence_positions':
{'Start': 0.5, 'End': 0, 'Middle': 0.09999999999999998}, 'sentence_similarities': {'High': 0, 'Low': 1.0, 'Medium': 0.4}, 'nume
rical_score_ratios': {'High': 0, 'Low': 0.8338676600919719, 'Medium': 0.5661323399080281}, 'thematic_word_feature_scores': {'Hi
gh': 0, 'Low': 1.0, 'Medium': 0.4}, 'sentence_weights2': {'High': 0, 'Low': 0.618251867739593, 'Medium': 0.781748132260407}},
'sentence4': {'title_feature_scores': {'High': 0, 'Low': 1.0, 'Medium': 0.4}, 'sentence_lengths': {'Long': 1, 'Short': 0, 'Medi
um': 0}, 'term_weights': {'High': 0.9999999999999996, 'Low': 0, 'Medium': 0}, 'sentence_positions': {'Start': 0.25, 'End': 0,
'Middle': 0.35}, 'sentence_similarities': {'High': 0, 'Low': 1.0, 'Medium': 0.4}, 'numerical_score_ratios': {'High': 0, 'Low':
0.859267849893325, 'Medium': 0.540732150106675}, 'thematic_word_feature_scores': {'High': 0, 'Low': 1.0, 'Medium': 0.4}, 'sente
nce_weights2': {'High': 0, 'Low': 0.051338587005612, 'Medium': 0.651338587005612}}, 'sentence5': {'title_feature_scores': {'Hig
h': 0, 'Low': 1.0, 'Medium': 0.4}, 'sentence_lengths': {'Long': 1, 'Short': 0, 'Medium': 0}, 'term_weights': {'High': 0.9999999
999999996, 'Low': 0, 'Medium': 0}, 'sentence_positions': {'Start': 0, 'End': 0, 'Middle': 0.6}, 'sentence_similarities': {'Hig
h': 0, 'Low': 1.0, 'Medium': 0.4}, 'numerical_score_ratios': {'High': 0, 'Low': 0.8214281034995066, 'Medium': 0.578571896500493
5}, 'thematic_word_feature_scores': {'High': 0, 'Low': 1.0, 'Medium': 0.4}, 'sentence_weights2': {'High': 0, 'Low': 0.177585889
86476276, 'Medium': 0.7775858898647627}}, 'sentence6': {'title_feature_scores': {'High': 0, 'Low': 1.0, 'Medium': 0.4}, 'senten
ce_lengths': {'Long': 1, 'Short': 0, 'Medium': 0}, 'term_weights': {'High': 0.9999999999999996, 'Low': 0, 'Medium': 0}, 'senten
ce_positions': {'Start': 0, 'End': 0.25, 'Middle': 0.85}, 'sentence_similarities': {'High': 0, 'Low': 1.0, 'Medium': 0.4}, 'num
erical_score_ratios': {'High': 0, 'Low': 0.8546445948767187, 'Medium': 0.5453554051232814}, 'thematic_word_feature_scores': {'H
igh': 0, 'Low': 1.0, 'Medium': 0.4}, 'sentence_weights2': {'High': 0, 'Low': 0.1831926155125001, 'Medium': 0.783192615512500
```

consequence based on the fuzzy logic principles. The fuzzy rules define the relationships between the linguistic variables of the features, and the fuzzy values represent the degree of membership of each linguistic variable for a particular feature in a sentence. By evaluating the fuzzy rules with the fuzzy values, the function determines the appropriate consequence label for the given sentence.

Rule Base: Once the scores have been fuzzified into linguistic terms, fuzzy rules are applied to determine the importance of each sentence. Fuzzy rules are predefined statements that relate the input (linguistic terms) to the output (importance level) based on expert knowledge or predefined heuristics.

The fuzzy rules can take various forms and depend on the specific context and requirements of the text summarization system. These rules can be defined using IF-THEN statements or other logical expressions. For instance, a fuzzy rule might state: IF (NoWordInTitle is VH) and (SentenceLength is H) and (TermFreq is VH) and (SentencePosition is H) and (SentenceSimilarity is VH) and (NoProperNoun is H) and (NoThematicWord is VH) and (NumbericalData is H) THEN (Sentence is important)

By applying the fuzzy rules to the fuzzified scores, the system assigns an importance level (unimportant, average, or important) to each sentence.

Fuzzy Inference:

a. Apply the fuzzy rules to determine the overall importance of each sentence based on the fuzzy inputs.

b. Use fuzzy inference methods such as Mamdani or Sugeno to aggregate the fuzzy outputs.

In the fuzzy inference system, we have used the calculateConsequence function to determine the output/consequence of each fuzzy rule: Here's how the 'calculateConsequence' function works:

1. The 'fuzzyRules' parameter represents a list of fuzzy rules. A fuzzy rule consists of an antecedent and a consequence. The antecedent describes the conditions or inputs, while the consequence represents the output or result.

2. The 'fuzzyValues' parameter is a dictionary(in our case of code) that maps fuzzy variables to their corresponding membership values. These membership values indicate the degree of membership or truthfulness of a linguistic variable for a particular feature.

fuzzyValues =
'sentence1':
'titleFeatureScores':

'Low': 0.2,
'Medium': 0.6,
'High': 0.8
,
'sentenceLengths':
'Short': 0.4,
'Medium': 0.7,
'Long': 0.5
,
,
'sentence2':
'titleFeatureScores':
'Low': 0.3,
'Medium': 0.5,
'High': 0.7
,
'sentenceLengths':
'Short': 0.6,
'Medium': 0.4,
'Long': 0.8
,
.... Similarly for other features too

3. The function takes the generated fuzzy rules (‘fuzzyRules‘) and fuzzy values (‘fuzzyValues‘) as inputs. It initializes an empty dictionary called ‘consequenceValues‘. This dictionary will store the calculated consequence values for each fuzzy rule

4. It iterates over each fuzzy rule in the ‘fuzzyRules‘ list.

5. For each fuzzy rule, it evaluates the antecedents by extracting the feature and linguistic value from each antecedent string. It looks up the corresponding fuzzy value from the ‘fuzzyValues‘ dictionary.

6. It checks if all the antecedent results are greater than 0. If they are, it means the fuzzy rule matches the fuzzy values.

7. If the fuzzy rule matches, it retrieves the consequence from the fuzzy rule and calculates the minimum of the antecedent results.

8. It updates the ‘consequenceValues‘ dictionary with the calculated consequence value. If the consequence already exists in the dictionary, it takes the maximum value

between the existing value and the new value.

9. After evaluating all the fuzzy rules, it checks if there are any calculated consequence values in the ‘consequenceValues‘ dictionary.

10. If there are, it finds the maximum consequence value and retrieves the corresponding consequence which will act as a sentence score.

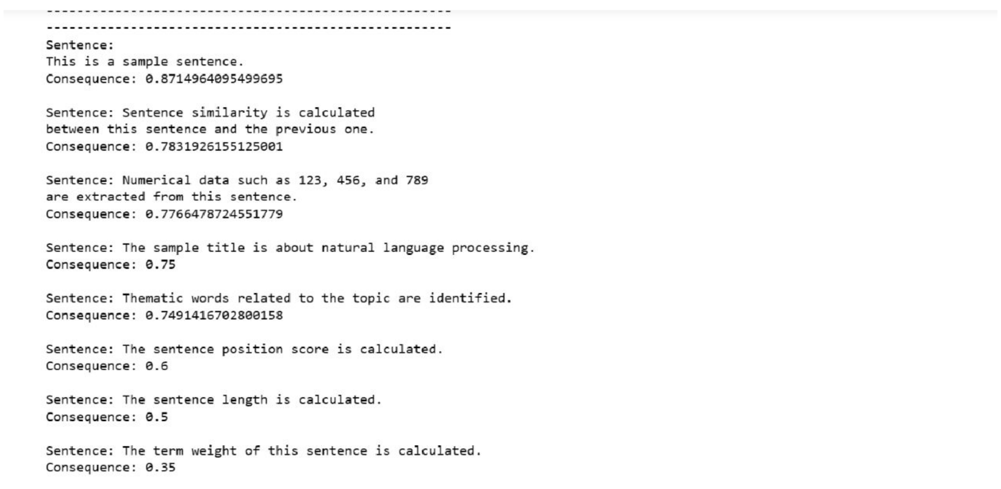
11. Finally, it returns the determined consequence. If none of the fuzzy rules match, it returns the default consequence of ‘’Unknown’‘.

In summary, the consequence value represents the importance level of the sentence, and it is determined by finding the maximum of the minimum antecedent values among the applicable fuzzy rules. The minimum value among these fuzzy values indicates the weakest or least-supported condition among the antecedents.In fuzzy logic, taking the minimum of the antecedent results is a common approach to model the logical operator "AND" in fuzzy rules. The minimum operation ensures that the overall degree of compatibility or support for the consequence is determined by the weakest condition among the antecedents, reflecting the principle of fuzzy logic that "a chain is only as strong as its weakest link."

Defuzzification:

a. Convert the fuzzy output into a crisp value representing the degree of importance for each sentence.

b. Use defuzzification techniques such as centroid, mean of maxima, or weighted average. But in our code, we have just calculated the importance of each sentence using the consequence value returned by our function which will act as a sentence core so we haven’t performed defuzzification in the net.



```
----------------------------------------------------
----------------------------------------------------
Sentence:
This is a sample sentence.
Consequence: 0.8714964095499695

Sentence: Sentence similarity is calculated
between this sentence and the previous one.
Consequence: 0.7831926155125001

Sentence: Numerical data such as 123, 456, and 789
are extracted from this sentence.
Consequence: 0.7766478724551779

Sentence: The sample title is about natural language processing.
Consequence: 0.75

Sentence: Thematic words related to the topic are identified.
Consequence: 0.7491416702800158

Sentence: The sentence position score is calculated.
Consequence: 0.6

Sentence: The sentence length is calculated.
Consequence: 0.5

Sentence: The term weight of this sentence is calculated.
Consequence: 0.35
```

Sentence Selection:

a. Rank the sentences based on their importance scores obtained from defuzzification.

b. Select the top-ranked sentences to form the summary.

Post-processing:

a. Concatenate the selected sentences to generate the final summary.

b. Perform any necessary formatting or cleanup.

2.1.2 Text Summarization using Feature weights by Fuzzy

Preprocessing: Text representation: Fuzzification: Above three steps will be calculated similarly to the fuzzy inference method Weighting of features: In this method, the ‘featureWeights‘ dictionary assigns weights to different features that are used to calculate the scores for each sentence. The purpose of these weights is to control the importance of each feature in the overall sentence ranking process. Let’s go through each feature and discuss why some features are weighted more and others less:

1. ‘titleFeatureScores‘ (Weight: 0.5): This feature probably represents how well a sentence aligns with the main title of the text or the summary. Assigning it a higher weight of 0.5 suggests that the model considers the title’s relevance to be significant in determining the importance of sentences.

2. ‘sentenceLengths‘ (Weight: 0.3): The length of a sentence can sometimes be an indicator of its importance. By giving it a weight of 0.3, the model acknowledges that sentence length plays a role, but it’s not as crucial as other features.

3. ‘termWeights‘ (Weight: 0.2): Term weights refer to the importance of specific keywords or terms in a sentence. With a weight of 0.2, the model considers the presence of important keywords, but it’s not the most critical factor.

4. ‘sentencePositions‘ (Weight: 0.4): Sentence positions may indicate the beginning or ending of a paragraph, section, or document. By assigning a weight of 0.4, the model values the position of a sentence but not more than some other features.

5. ‘sentenceSimilarities‘ (Weight: 0.6): Sentence similarity measures how similar a sentence is to other sentences in the text. Giving it a weight of 0.6 indicates that the model considers sentence similarity to be a strong indicator of sentence importance.

6. ‘numericalScoreRatios‘ (Weight: 0.3): This feature might represent the ratio of numerical scores associated with the sentences. With a weight of 0.3, the model considers

it important, but not as much as some other features.

7. ‘thematicWordFeatureScores‘ (Weight: 0.5): Thematic word feature scores likely refer to the presence of thematic words in a sentence. Assigning a weight of 0.5 indicates that thematic words play a significant role in determining sentence importance.

8. ‘sentenceWeights‘ (Weight: 0.4): This is another feature related to sentence weights, and it’s given a weight of 0.4. it represent a more sophisticated measure related to sentence importance.

Overall, the distribution of weights depends on the specific domain, dataset, and requirements of the extractive text summarization task. By assigning different weights to various features, the model can be fine-tuned to produce summaries that align with the desired characteristics or priorities of the summarization process. Let’s compare the weights of different features and discuss potential reasons for their specific values -

1. *titleFeatureScores vs. thematicWordFeatureScores :*Both of these features can be given approximately equal weights, suggesting that the model considers both the alignment with the main title and the presence of thematic words in sentences as somewhat equally important in determining sentence scores.TitleFeatureScores may help to ensure that sentences align well with the main topic or theme of the text, while thematicWordFeatureScores emphasize the significance of thematic keywords or words relevant to the subject matter.we can calculate thematic word features using text(rather than title) so this can have less or more weight to the title feature score.

2. *sentenceSimilarities vs. sentencePositions :* so we have presumably assigns a higher weight to sentence similarities, indicating that the similarity of a sentence to other sentences in the text is considered more critical than the position of the sentence within the text. This is basically driven by the assumption that sentences that are more similar to others are more likely to contain essential information and capture the core concepts of the text.

3. *sentenceLengths vs. numericalScoreRatios :* sentence lengths can be considered more significant than the numericalScoreRatios because we need small-sized summaries. But when we are extracting important points from a text both the features can have the same weight, implying that the model perceives sentence length and numerical score ratios as equally significant. SentenceLengths can influence importance, as shorter sentences may be preferred in a summary, while numericalScoreRatios might represent some metric related to sentence scores that affect importance rankings.

It’s important to note that the choice of feature weights heavily depends on the characteristics of the data, the specific goals of the summarization task, and the desired trade-offs between different aspects of summarization (e.g., relevance, diversity, length). These weights are often determined through experimentation and fine-tuning, where the model's performance is evaluated on a validation set or against human-generated summaries to find the best configuration.

Sentence score calculation- To calculate the sentence score(for a single sentence)- 1. we have used the max membership method .it will extract the maximum membership value among the low, medium, or high sets for a particular feature that corresponds to a sentence. 2. then the extracted membership value will be multiplied by the respective feature. 3. then we will do the above steps for each feature of the given sentence and after that, we will add these final values for all the features to calculate the overall sentence score.

Once we are done with the sentence score calculation for each sentence we will then do sentence selection similar to the fuzzy inference method.

2.1.3 Optimization of Feature Weights

To optimize the feature weights, we used a genetic algorithm. We used the real-coded genetic algorithm, not the binary one, because we have real values. We don't want to convert them into binary and then back into decimal values again, as we need the real-coded values to calculate the fitness. Also, if we use binary code, we will have long string lengths and will face the problem of Hamming cliffs, where we need to change a lot of bits just to go to the next value for a particular feature.

We have employed the f-score as our fitness function, which compares a predetermined gold summary with a model summary. The model summary is generated iteratively by opti- mizing feature values. Population initialization can be random or heuristic-driven, as we know about feature importance. We have used the following methods for each step:

• Parent selection: A 2-tournament selection mechanism is used to select individuals from the current population to become parents for the next generation. It forms the mating pool. The basic idea is to randomly select a small subset of individuals (the tournament) from the population and then choose the best individual from that subset to be a parent. This process is repeated until a sufficient number of parents are selected for the next generation. The key parameter in tournament selection is the tournament size, which determines how many individuals participate in each tournament. In this case, we use a tournament size of 2.

Crossover: Simulated binary crossover (SBX) is used because naive crossover operators, such as the single- point crossover, may fail to perform well. This is because naive crossover operators only search within the current values of the decision variables. As a result, they even- tually depend on the mutation operator to generate new values for the decision variables. This is why simulated binary crossover is preferred. SBX simulates the single- point crossover on binary strings. It requires two parents to generate two offspring. The two offspring have a sepa- ration proportional to the parents. Crossover is performed with a high probability.

• Mutation: The algorithm uses environment selection. We combine the generated offspring from the mutation and initial population and then arrange them according to their fitness values. Since our problem is maximization, we will select the top individuals, the same size as the initial population. These selected individuals will then become the new initial population of the next generation. This will help us to optimize our features and values.

Algorithm-1 and Algorithm-2 describe the process of cross- over and mutation used in the fuzzy-weight-based method.

Procedure for SBX Crossover

Inputs are P,D,Pc,ηc.

- Randomly select a pair of parents (P'_a and P'_b) from mating pool.
- Generate a random number(r) between 0 and 1.
- If r >= P_c then copy the parent solutions as offspring.
- If r < P_c generate D random numbers (u) for each variable.
- Determine ß of each variable.
- Generate two offspring (O_a and O_b) using

$$O_a = 0.5[(1+\beta)P'_a + (1-\beta)P'_b]$$
$$O_b = 0.5[(1-\beta)P'_a + (1+\beta)P'_b]$$

$$\beta = (2u)^{1/(\eta_c+1)} \quad \text{if } u \leq 0.5$$
$$(1/2(1-u))^{1/(\eta c+1)} \quad \text{other wise}$$

Procedure for Polynomial Mutation

Inputs are P,D,Pc,ηc.

- Randomly select a pair of parents (P'_a and P'_b) from mating pool.
- Generate a random number(r) between 0 and 1.
- If r >= P_c then copy the parent solutions as offspring.
- If r < P_c generate D random numbers (u) for each variable.
- Determine ß of each variable.
- Generate two offspring (O_a and O_b) using

$$O_a = 0.5[(1+\beta)P'_a + (1-\beta)P'_b]$$
$$O_b = 0.5[(1-\beta)P'_a + (1+\beta)P'_b]$$

$$\beta = (2u)^{1/(\eta_c+1)} \quad \text{if } u \leq 0.5$$
$$(1/2(1-u))^{1/(\eta c+1)} \quad \text{other wise}$$

2.1.4 Comparison

Fuzzy rules method v/s fuzzy weighted method

The main difference between the two methods lies in their application of fuzzy logic:

• In the first method, fuzzy rules guide the summarization decisions.

• In the second method, fuzzy logic is used to handle uncertainty in the feature values and determine the overall sentence scores.

The second method is easier to implement in comparison to fuzzy rules but it involves more human involvement like in the calculation of feature weight which requires great domain knowledge.

It also requires the user to decide which membership function is used,along with linguistic variables etc for each feature like the fuzzy rules.

We have used first method because it is calculating the sentence score with less human involvement.

2.2 Abstractive Summarization

Generally, text summarization using an Abstractive approach involves the following necessary steps-

Preprocessing :

Text Cleaning: Remove special characters, symbols, and punctuation marks. Convert the text to lowercase for case insensitivity. Remove extra spaces and tabs. Handling special cases like contractions,abbreviations etc.
Tokenization: Break the text into individual words or tokens and assign unique IDs to each token. Then a mapping is created from tokens to numerical IDs in which vocabulary should cover all unique tokens in the dataset. This step is essential for further processing and analysis.
Selecting a model: Some pre-trained transformer-based models for summarizing the text are BART,T5 or GPT-3. We chose the BART model. For abstract summarization, it consists of an encoder-decoder architecture. As we are using a pre-trained model, we need to perform transfer learning on the model using the training data to adapt it to do the summarization. Use the trained model to generate a summary, this is done by passing the input through the models decoder, producing the summarized output.
Post- processing: After the summary is generated, remove unnecessary tokens and apply any additional post-processing steps to ensure a coherent and grammatically correct summary.

1. We have imported libraries like torch, BartTokenizer and BartForConditionalGeneration from transformers.

2. We loaded the pre-trained BART model and tokenizer provided by the Hugging Face Transformers library.‘facebook/bart-large-cnn‘is the model which is specifically trained for summarization tasks.

3. Defining Text Summarization Function

4. Here, input document is being tokenized by BART Tokenizer. The ‘encode‘ method converts the input text into token IDs, and ‘returnTensors=’pt’‘ ensures the return type is PyTorch tensors.

5. The ‘generate‘ method is used to generate the summary. Here, we pass the tokenized input as ‘inputs‘, ‘numBeams=4‘ specifies the number of beams to use during decoding (beam search), ‘maxLength=100‘ sets the maximum length of the summary, and ‘earlyStopping=True‘ stops the generation process when the model predicts an end-of-sequence token.

6. The generated summary is in token IDs format, so we use the tokenizer to decode it back into a readable text format. The ‘squeeze()‘ method is used to remove any unnecessary dimensions, and ‘skipSpecialTokens=True‘ removes special tokens like ‘[CLS]‘, ‘[SEP]‘, etc., from the final summary.

7. The function ‘generateSummary(document)‘ takes an input document as a string and returns the abstractive summary of the document using the pre-trained BART model. The summary is generated by first tokenizing the input, then using the BART model for text generation with beam search, and finally decoding the generated token IDs back into a human-readable summary.

CHAPTER 3

RESULTS AND IMPROVEMENTS

3.1 Results

3.1.1 Result for Text Summarization using Fuzzy Rules

As we can see once our consequence function will return the consequence of each sentence which will act as a sentence weight, we will arrange the sentences on the basis of sentence weights in their decreasing order. After that whatever number of lines will be required by the user for their summary, we will pick the same number of top sentences and then we will again arrange them in the same order as they were in the text .just to generate a summary we concatenate them into a paragraph.

3.1.2 Result for Text Summarization using Feature Weights by Fuzzy

Similar to the fuzzy rules base text summarization it will also eventually gives us the calculated summary based on sentence scores for each sentence. It is faster than the previous method though for more complex summaries with the involvement of more different types of data, previous method previous method may be more suitable since it takes every possibility into account.

3.1.3 Evaluation

We used the F-score measure to evaluate the performance of the presented model. The F-score is a combination of precision and recall, and provides a balanced measure of the summarization system's performance. It is defined as follows.
1. True Positives (TP): The number of words (such as sentences or phrases) in the reference summary that are also present in the generated summary.

2.False Positives (FP): The number of words in the generated summary that are not present in the reference summary.

3. False Negatives (FN): The number of words in the reference summary that are missing from the generated summary.
4. Precision: It measures how many generated words are actually relevant or correct, out of the total generated words. It is calculated as given in Eq-1.

$$Precision = \frac{TP}{TP + FP} \tag{3.1}$$

5. Recall: It measures how many relevant or correct words from the reference summary were captured by the generated summary. It is calculated as given in Eq-2.

$$Recall = \frac{TP}{TP + FN} \tag{3.2}$$

6. F-score: It is the harmonic mean of precision and recall. It is calculated as defined in Eq-3.

$$F - score = \frac{2 * (Precision * Recall)}{Precision + Recall} \tag{3.3}$$

To calculate these metrics for text summarization, we need a reference summary (also called a gold summary) and the generated summary produced by our summarization system. We compare the content of the generated summary to the reference summary to count true positives, false positives, and false negatives. We then use these counts to calculate precision, recall, and F-score. Our dataset is a collection of random texts of 20-line online paragraphs. Their human-generated summaries will act as gold summaries. We have shown the F-scores for the three models discussed in the paper. As we can see, the fuzzy inference model using fuzzy rules is much better than the other two. Although we have optimized the fuzzy method using feature weights, we cannot achieve the maximum F-score due to early convergence.

Table 3.1: Results of the presented fuzzy-based models for extractive text summarization

Measure	TF-IDF	Fuzzy (using feature weights)	Fuzzy (using fuzzy rules)
F-scores	0.20	0.42	0.44
F-scores	0.54	0.56	0.66
F-scores	0.24	0.42	0.58

3.2 Improvements

In the text summarization based on fuzzy rules, we can improve it further in the following ways-

1. By adding new features in the text summarization.

2. Assigning the appropriate membership functions for each feature according to the domain.

3. Defining different linguistic variable sets for different features.

4. Also while defining the boundaries of the membership functions we can be more precise and respective according to each feature.

In the text summarization-based feature weights, we can improve it further in the following ways-

1. We can optimize it further using the methods same as discussed above.

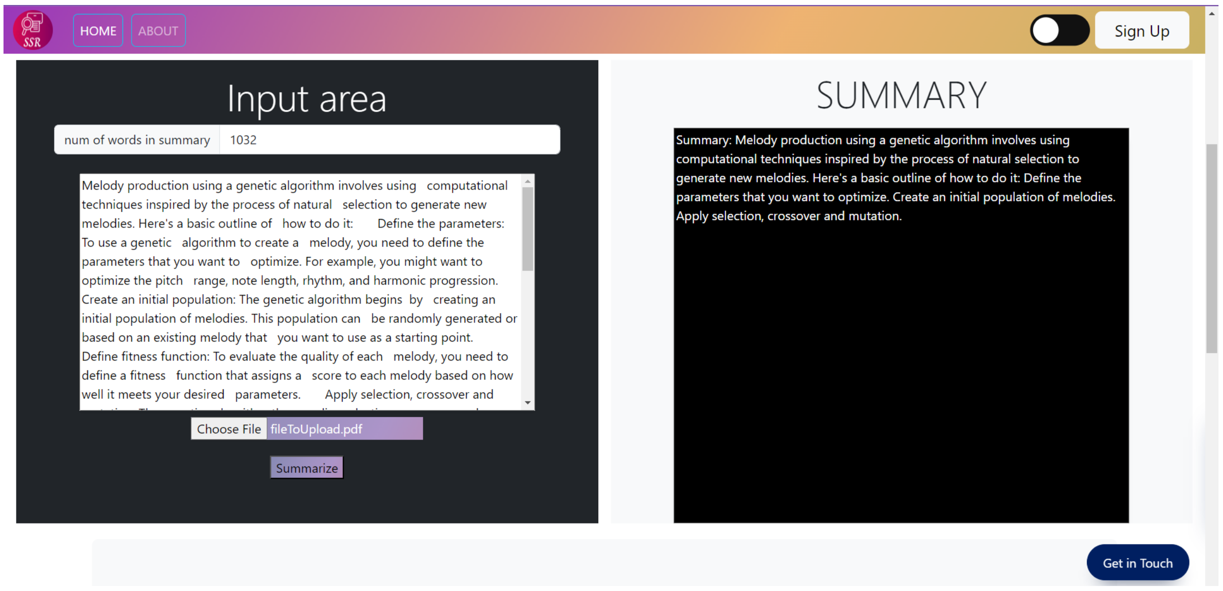
2. Apart from that most importantly we can optimize the feature weights.

3. By assigning different weights to various features, the model can be fine-tuned to produce summaries that align with the desired characteristics or priorities of the summarization process.

3.3 Key Features of Website

1.Text Input:

Users can enter a text or document of text they want to summarize.With this feature,



num of words in summary 1032

Melody production using a genetic algorithm involves using computational techniques inspired by the process of natural selection to generate new melodies. Here's a basic outline of how to do it: Define the parameters: To use a genetic algorithm to create a melody, you need to define the parameters that you want to optimize. For example, you might want to optimize the pitch range, note length, rhythm, and harmonic progression. Create an initial population: The genetic algorithm begins by creating an initial population of melodies. This population can be randomly generated or based on an existing melody that you want to use as a starting point. Define fitness function: To evaluate the quality of each melody, you need to define a fitness function that assigns a score to each melody based on how well it meets your desired parameters. Apply selection, crossover and

fileToUpload.pdf

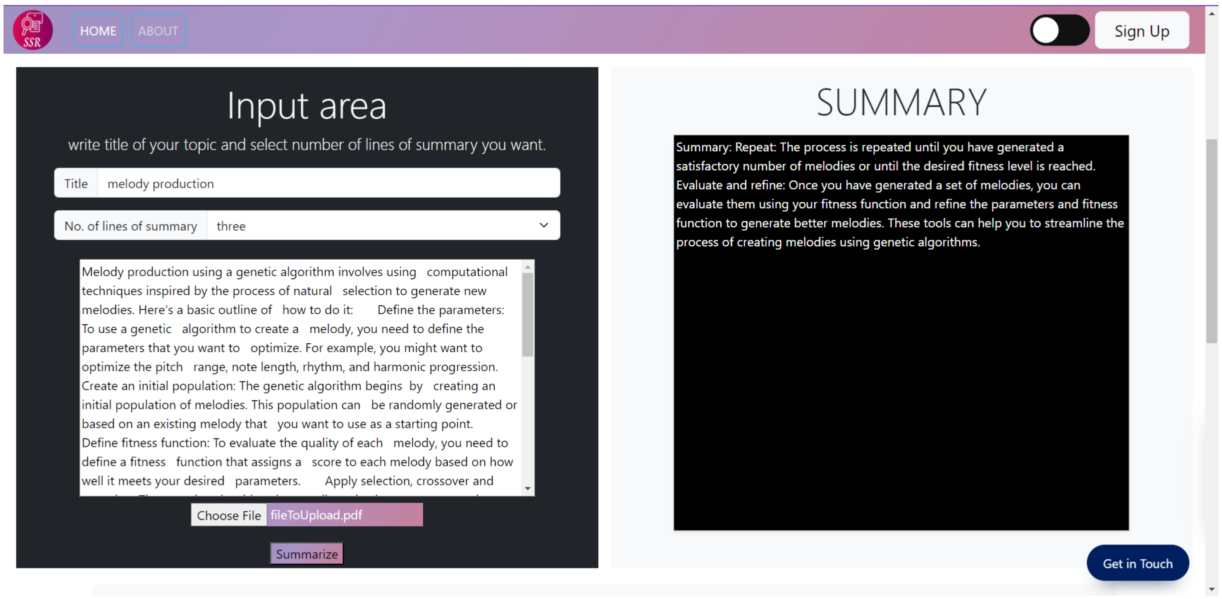
Summary: Melody production using a genetic algorithm involves using computational techniques inspired by the process of natural selection to generate new melodies. Here's a basic outline of how to do it: Define the parameters that you want to optimize. Create an initial population of melodies. Apply selection, crossover and mutation.

Figure 3.1: Abstractive Summarization.

users can easily provide the content they want to summarize.

2.Sentence Count Selection:

The user can choose a limit on the number of sentences required for summarization. This feature allows the user to control the length of the generated synopsis according



melody production

three

Melody production using a genetic algorithm involves using computational techniques inspired by the process of natural selection to generate new melodies. Here's a basic outline of how to do it: Define the parameters: To use a genetic algorithm to create a melody, you need to define the parameters that you want to optimize. For example, you might want to optimize the pitch range, note length, rhythm, and harmonic progression. Create an initial population: The genetic algorithm begins by creating an initial population of melodies. This population can be randomly generated or based on an existing melody that you want to use as a starting point. Define fitness function: To evaluate the quality of each melody, you need to define a fitness function that assigns a score to each melody based on how well it meets your desired parameters. Apply selection, crossover and

fileToUpload.pdf

Summary: Repeat: The process is repeated until you have generated a satisfactory number of melodies or until the desired fitness level is reached. Evaluate and refine: Once you have generated a set of melodies, you can evaluate them using your fitness function and refine the parameters and fitness function to generate better melodies. These tools can help you to streamline the process of creating melodies using genetic algorithms.

Figure 3.2: Extractive Summarization.

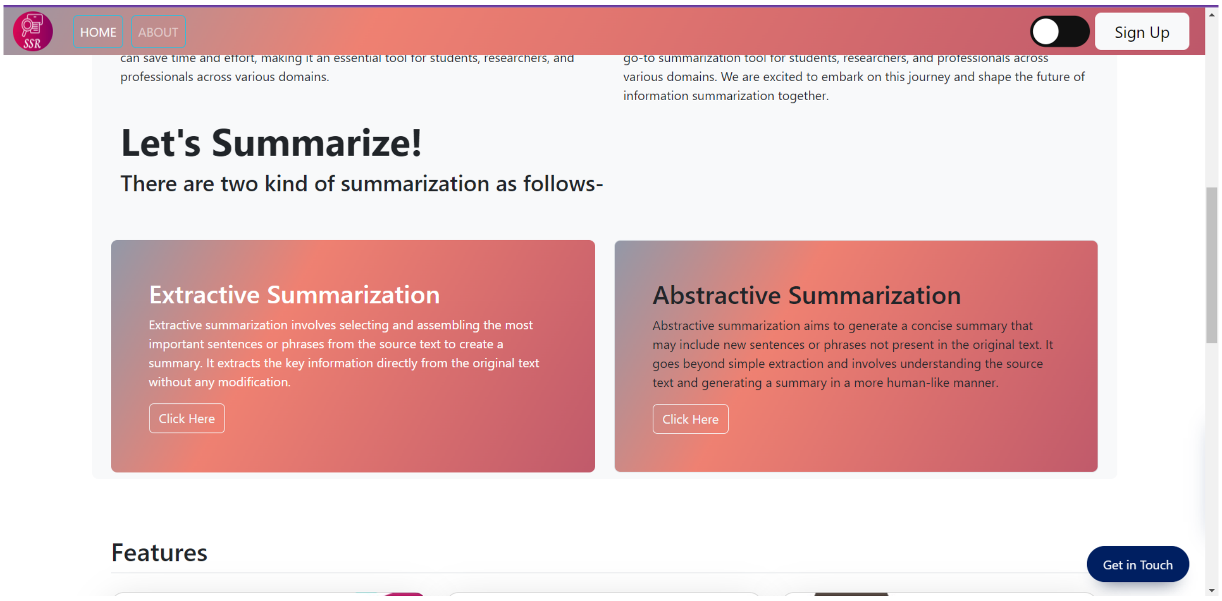
to their preferences and requirements.

3.Auto Summarization:

This website uses advanced natural language processing algorithms to automatically generate a summary of your input text. This feature allows users to quickly and easily get an overview of long articles and documents.

4.Both Extract and abstractive Summary:

We used both approaches.Users can choose their preferences.Either method ensures



can save time and effort, making it an essential tool for students, researchers, and professionals across various domains.

go-to summarization tool for students, researchers, and professionals across various domains. We are excited to embark on this journey and shape the future of information summarization together.

Let's Summarize!

There are two kind of summarization as follows-

Extractive Summarization

Extractive summarization involves selecting and assembling the most important sentences or phrases from the source text to create a summary. It extracts the key information directly from the original text without any modification.

Abstractive Summarization

Abstractive summarization aims to generate a concise summary that may include new sentences or phrases not present in the original text. It goes beyond simple extraction and involves understanding the source text and generating a summary in a more human-like manner.

that the summary contains the important information contained in the input

5.Interactive User Interface:

This website offers an intuitive and user-friendly interface for a smooth user experience. Users can easily navigate your site and get an overview with just a few clicks

6.Efficiency:

The summarization process is optimized for efficiency, providing users with a efficient summary even for long articles.

7 .Responsive Design:

The website is designed to be responsive, accessible on different devices such as desktops, tablets and smartphones, and looks great.

8.Subscribe to updates:

This site provides a subscription option for free that allows users to opt-in to receive information by email. Email notification: After subscribing, the user will receive periodic email alerts with updates as a newsletter.

CHAPTER 4

CONCLUSION

In conclusion, the fuzzy system method employed in this text summarization offers a valuable and effective approach for processing and condensing information. By leveraging fuzzy logic to handle linguistic uncertainties and approximate reasoning, the summarization process becomes more adaptive and flexible. Through the extraction of key features and linguistic patterns, the method successfully captures the essence of the original text and presents a concise summary that retains essential information.

The fuzzy system's ability to handle vague and imprecise data enables it to deal with the inherent ambiguity in natural language, making it a suitable choice for text summarization tasks. Moreover, the method's performance can be optimized through appropriate parameter tuning and linguistic rule refinement, resulting in even more accurate and informative summaries.

However, while the fuzzy system method shows promise, it is essential to acknowledge that no summarization technique is perfect. Depending solely on fuzzy logic might have limitations in capturing very subtle nuances or complex relations present in some texts. Therefore, researchers and developers should continue exploring hybrid approaches that combine fuzzy systems with other AI techniques to achieve more comprehensive and refined summaries.

Overall, the fuzzy system method proves to be a valuable tool for text summarization, and further advancements in this area could lead to significant improvements in automatic summarization algorithms, benefiting various applications in information retrieval, natural language processing, and knowledge extraction.

Luhn (1958) Lin (2004) Fattah and Ren (2008)

REFERENCES

[1] Fattah, M. A. and Ren, F.: 2008, Automatic text summarization, *Proceedings of World Academy of Science, Engineering and Technology*, Vol. 27, pp. 192–195.

[2] Lin, C.-Y.: 2004, ROUGE: A package for automatic evaluation of summaries, *Proceedings of Workshop on Text Summarization of ACL*, Spain.

[3] Luhn, H. P.: 1958, The automatic creation of literature abstracts, *IBM Journal of Research and Development* **2**, 159–165.