

Natural Language Processing (NLP)

CA2 PROJECT

Extractive Text Summarization for Hindi & Malayalam

Submitted By – Sharon Philip – 6961

1) Introduction:

Extractive text summarization is a technique in Natural Language Processing (NLP) that involves selecting key sentences or phrases directly from a given text to create a concise and meaningful summary. Unlike abstractive summarization, which generates new sentences, extractive summarization retains the original wording and structure from the source text. This technique is widely used in applications such as news summarization, document condensation, and automated content generation.

The goal of extractive summarization is to identify the most informative parts of a document while preserving coherence and meaning. It plays a crucial role in handling vast amounts of textual data, allowing users to quickly grasp essential information from large documents, research papers, or news articles.

2) Key Components of Extractive Text Summarization:

- ❖ **Text Input:** User-provided text that needs to be summarized.
- ❖ **Tokenization:** Splitting text into words and sentences.
- ❖ **Stop word Removal & Punctuation Filtering:** Removing unnecessary words to focus on key content.
- ❖ **Word Frequency Calculation:** Identifying important words based on their occurrence in the text.
- ❖ **Sentence Scoring:** Assigning scores to sentences based on word importance.
- ❖ **Sentence Selection:** Choosing the highest-scoring sentences for the summary.
- ❖ **GUI (Tkinter):** Providing an interface for users to input text and receive a summary.

3) Techniques Used in Extractive Text Summarization:

Luhn's Algorithm (Frequency-Based Sentence Scoring) – Detailed Explanation

Luhn's Algorithm is one of the earliest and simplest techniques for extractive text summarization. It was developed by Hans Peter Luhn in 1958 and assumes that words with high frequency in a document carry significant meaning, while common words (like "the," "is," "and") provide little useful information.

How Luhn's Algorithm Works The algorithm follows these key steps:

1. Preprocessing the Text:

- Tokenization: Split the document into sentences and words.
- Remove stop words (common words such as "the," "is," "and").
- Remove punctuation and convert text to lowercase.

2. Word Frequency Calculation:

- Compute the frequency of each word in the text.
- Sort words by frequency, identifying the most frequent and important words.

3. Identifying "High-Information" Words:

- Define a cutoff threshold based on word frequency.
- Words with frequency above the threshold are considered important, while those below are ignored.

4. Sentence Scoring:

- Identify sentences containing a high density of important words.
- Assign scores to sentences based on the number of important words they contain and their proximity to each other.

5. Selecting Sentences for Summary:

- Choose the highest-scoring sentences while maintaining the document's coherence.

4) Code:

```
✓ 0s text = """"नकारात्मकता से भरी दुनिया में, दयालुता और करुणा की शांति
```

```
✓ 0s [2] len(text)
```

```
⇌ 1287
```

```
✓ 20s [3] import spacy
      from spacy.lang.en.stop_words import STOP_WORDS
      from string import punctuation
```

```
✓ 1s [4] nlp = spacy.load('en_core_web_sm')
```

```
✓ 0s [5] doc = nlp(text)
```


```
✓ 0s [6] tokens = [token.text.lower() for token in doc
               if not token.is_stop and
               not token.is_punct and
               token.text != '\n']
```

```
[7] tokens
```

```
⇌ 'संस्कृति',
  'विकसित',
  'कर',
  'सकते',
  'हैं',
  'जो',
  'हमारे',
  'और',
  'हमारे',
  'आस',
  'पास',
  'के',
```

```
▶ tokens1=[]
  stopwords = list(STOP_WORDS)
  allowed_pos = ['ADJ', 'PROPN', 'VERB', 'NOUN']
  for token in doc:
      if token.text in stopwords or token.text in punctuation:
          continue
      if token.pos_ in allowed_pos:
          tokens1.append(token.text)
```


 tokens1

 ['नकारात्मकता',
'से',
'भरी',
'दुनिया',
'में',
'दयालुता',
'और',
'करुणा',
'की',
'को',
'याद',
'रखना',
'है',
'।',
'के',

```
[10] from collections import Counter
```

```
[11] word_freq = Counter(tokens)
```


```
[12] word_freq
```

 {'स्तर': 1,
'पहल': 1,
'लेकर': 1,
'वैश्विक': 1,
'आंदोलनों': 1,
'लोग': 1,
'सामाजिक': 1,
'पर्यावरणीय': 1,

```
[13] max_freq = max(word_freq.values())
```

```
[14] max_freq
```

 11

```
 for word in word_freq.keys():  
    word_freq[word] = word_freq[word]/max_freq
```

[16] word_freq



```
'स्तर': 0.09090909090909091,  
'पहल': 0.09090909090909091,  
'लेकर': 0.09090909090909091,  
'वैश्विक': 0.09090909090909091,  
'आंदोलनों': 0.09090909090909091,  
'लोग': 0.09090909090909091,  
'सामाजिक': 0.09090909090909091,  
'पर्यावरणीय': 0.09090909090909091,  
'मुद्दों': 0.09090909090909091,  
'निपटने': 0.09090909090909091,  
'एकजुट': 0.09090909090909091,
```

[17] sent_token = [sent.text for sent in doc.sents]



sent_token



```
['नकारात्मकता से भरी दुनिया में, दयालुता और करुणा की शक्ति को याद रखना महत्वपूर्ण है। दयालुता',  
'के छोटे-छोटे काम किसी के दिन को खुशनुमा',  
'बना सकते हैं, उसका मनोबल बढ़ा सकते',  
'हैं और सकारात्मकता की लहर पैदा कर सकते हैं जो दूर-दूर तक फैल सकती है। चाहे वह किसी अजनबी को मुस्कुराना हो, ज़रूरत में किसी दोस्त की मदद करना हो या रि',  
'का हर काम किसी के जीवन में बदलाव लाने',  
'की क्षमता रखता है। व्यक्तिगत कार्यों से परे, सकारात्मक बदलाव लाने',  
'के लिए सामूहिक प्रयासों में भी अपार शक्ति',  
'होती है। जब समुदाय एक-दूसरे का समर्थन करने के लिए एक साथ आते हैं, तो अविश्वसनीय चीजें हो सकती हैं। ज़मीनी स्तर की पहल से लेकर वैश्विक आंदोलनों तक,',  
'लोग सामाजिक और पर्यावरणीय',  
'मुद्दों से निपटने',  
'के लिए एकजुट हो रहे हैं, सार्थक प्रगति कर रहे हैं और बेहतर भविष्य की उम्मीद जगा',  
'रहे हैं। हममें से हर एक के भीतर छिपी ताकत को पहचानना भी महत्वपूर्ण है। हम सभी में सकारात्मक प्रभाव डालने की क्षमता है, चाहे हमारे कार्य कितने',  
'भी छोटे क्यों न हों। अपनी सहज करुणा और सहानुभूति का उपयोग करके, हम दयालुता और सहानुभूति की संस्कृति विकसित कर सकते हैं जो हमारे और हमारे आस-पास के',  
'शक्ति को अपनाएँ, और एक समय में एक छोटा सा कार्य करके',  
'दुनिया को एक बेहतर जगह बनाने का प्रयास करें। साथ मिलकर, हम सभी के लिए एक उज्ज्वल, अधिक दयालु भविष्य बना सकते हैं।']
```

[19] sent_score = {}

```
for sent in sent_token:  
    for word in sent.split():  
        if word.lower() in word_freq.keys():  
            if sent not in sent_score.keys():  
                sent_score[sent] = word_freq[word]  
            else:  
                sent_score[sent] += word_freq[word]  
print(word)
```



कों
संस्कृति
विकसित
कर
सकते
हैं
जो
हमारे
और
हमारे
आस-पास
के

sent_score

```
('नकारात्मकता से भरी दुनिया में, दयालुता और करुणा की शक्ति को याद रखना महत्वपूर्ण है। दयालुता': 4.727272727272727,  
'के छोटे-छोटे काम किसी के दिन को खुशनुमा': 3.4545454545454546,  
'बना सकते हैं, उसका मनोबल बढ़ा सकते': 1.3636363636363638,  
'हैं और सकारात्मकता की लहर पैदा कर सकते हैं जो दूर-दूर तक फैल सकती है। चाहे वह किसी अजनबी को मुस्कुराना हो, ज़रूरत में किसी दोस्त की मदद करना हो या किसी सहकर्मी के लिए कोई विचारशील इशारा हो, दयालुता':  
12.000000000000005,  
'का हर काम किसी के जीवन में बदलाव लाने': 3.2727272727272727,  
'की क्षमता रखता है। व्यक्तिगत कार्यों से परे, सकारात्मक बदलाव लाने': 2.2727272727272725,  
'के लिए सामूहिक प्रयासों में भी अपार शक्ति': 2.8181818181818183,  
'होती है। जब समुदाय एक-दूसरे का समर्थन करने के लिए एक साथ आते हैं, तो अविश्वसनीय चीजें हो सकती हैं। जमीनी स्तर की पहल से लेकर वैश्विक आंदोलनों तक,'': 5.9999999999999999,  
'लोग सामाजिक और पर्यावरणीय': 1.0,  
'मुद्दों से निपटने': 0.6363636363636364,  
'के लिए एकजुट हो रहे हैं, सार्थक प्रगति कर रहे हैं और बेहतर भविष्य की उम्मीद जगा': 5.9999999999999999,  
'रहे हैं। हममें से हर एक के भीतर छिपी ताकत को पहचानना भी महत्वपूर्ण है। हम सभी में सकारात्मक प्रभाव डालने की क्षमता है, चाहे हमारे कार्य कितने': 7.1818181818181818,  
'भी छोटे क्यों न हों। अपनी सहज करुणा और सहानुभूति का उपयोग करके, हम दयालुता और सहानुभूति की संस्कृति विकसित कर सकते हैं जो हमारे और हमारे आस-पास के लोगों के जीवन को समृद्ध बनाती है। तो आइए दयालुता की  
शक्ति को अपनाएँ, और एक समय में एक छोटा सा कार्य करके': 16.909090909090914,  
'दुनिया को एक बेहतर जगह बनाने का प्रयास करें। साथ मिलकर, हम सभी के लिए एक उज्जवल, अधिक दयालु भविष्य बना सकते हैं।'': 5.9999999999999998)
```

```
[21] import pandas as pd
```

```
[22] pd.DataFrame(list(sent_score.items()),columns=['Sentence','Score'])
```



	Sentence	Score
0	नकारात्मकता से भरी दुनिया में, दयालुता और करुणा...	4.727273
1	के छोटे-छोटे काम किसी के दिन को खुशनुमा	3.454545
2	बना सकते हैं, उसका मनोबल बढ़ा सकते	1.363636
3	हैं और सकारात्मकता की लहर पैदा कर सकते हैं जो ...	12.000000
4	का हर काम किसी के जीवन में बदलाव लाने	3.272727
5	की क्षमता रखता है। व्यक्तिगत कार्यों से परे, स...	2.272727
6	के लिए सामूहिक प्रयासों में भी अपार शक्ति	2.818182
7	होती है। जब समुदाय एक-दूसरे का समर्थन करने के ...	6.000000
8	लोग सामाजिक और पर्यावरणीय	1.000000
9	मुद्दों से निपटने	0.636364
10	के लिए एकजुट हो रहे हैं, सार्थक प्रगति कर रहे ...	6.000000
11	रहे हैं। हममें से हर एक के भीतर छिपी ताकत को प...	7.181818
12	भी छोटे क्यों न हों। अपनी सहज करुणा और सहानुभू...	16.909091
13	दुनिया को एक बेहतर जगह बनाने का प्रयास करें। स...	6.000000



```
[23] from heapq import nlargest
```

```
[24] num_sentences =3  
n = nlargest(num_sentences,sent_score,key=sent_score.get)
```

```
 " ".join(n)
```

🔗 भी छोटे क्यों न हों। अपनी सहज करुणा और सहानुभूति का उपयोग करके, हम दयालुता और सहानुभूति की संस्कृति विकसित कर सकते हैं जो क्ति को अपनाएँ, और एक समय में एक छोटा सा कार्य करके हैं और सकारात्मकता की लहर पैदा कर सकते हैं जो दूर-दूर तक फैल सकती है। सहकर्मी के लिए कोई विचारशील इशारा हो, दयालुता रहे हैं। हममें से हर एक के भीतर छिपी ताकत को पहचानना भी महत्वपूर्ण है। हम सभी में

GUI Code

```
!pip install ipywidgets

import ipywidgets as widgets
from IPython.display import display
import spacy
from collections import Counter
from heapq import nlargest

# Load spaCy model once
nlp = spacy.load('en_core_web_sm')

def summarize_text(text):
    if not text.strip():
        return "⚠️ Please enter some text to summarize."

    # Tokenization and removing stopwords
    doc = nlp(text)
    tokens = [token.text.lower() for token in doc if not token.is_stop and not token.is_punct]

    # Calculating word frequency
    word_freq = Counter(tokens)
    if not word_freq:
        return "⚠️ No meaningful words found in the text."

    max_freq = max(word_freq.values())
    word_freq = {word: freq / max_freq for word, freq in word_freq.items()}

    # Sentence scoring
    sent_scores = {}
    sentences = list(doc.sents)

    for sent in sent.text.lower().split():
        if word in word_freq:
            sent_scores[sent.text] = sent_scores.get(sent.text, 0) + word_freq[word]

    # Select top 3 scoring sentences as summary
    summarized_sentences = nlargest(3, sent_scores, key=sent_scores.get)

    return " ".join(summarized_sentences)

# Create input widgets
text_input = widgets.Textarea(
    value="",
    placeholder='Enter text here',
    description='Text:',
    layout=widgets.Layout(width='600px', height='200px')
)
```



```

summarize_button = widgets.Button(
    ... description="Summarize",
    button_style="primary"
)

output_text = widgets.Textarea(
    value="",
    description='Summary:',
    layout=widgets.Layout(width='600px', height='150px'),
    disabled=True
)

# Function to summarize text when button is clicked
def on_summarize_clicked(b):
    summary = summarize_text(text_input.value)
    output_text.value = summary

# Attach event to button
summarize_button.on_click(on_summarize_clicked)

# Display the widgets
display(text_input, summarize_button, output_text)

```

```

Requirement already satisfied: ipywidgets in /usr/local/lib/python3.11/dist-packages (7.7.1)
Requirement already satisfied: ipykernel>4.5.1 in /usr/local/lib/python3.11/dist-packages (from ipywidgets) (6.17.1)
Requirement already satisfied: ipython-genutils<0.2.0 in /usr/local/lib/python3.11/dist-packages (from ipywidgets) (0.2.0)
Requirement already satisfied: traitlets>4.3.1 in /usr/local/lib/python3.11/dist-packages (from ipywidgets) (5.7.1)
Requirement already satisfied: widgetsnbextension<3.6.0 in /usr/local/lib/python3.11/dist-packages (from ipywidgets) (3.6.10)
Requirement already satisfied: ipython>4.0.0 in /usr/local/lib/python3.11/dist-packages (from ipywidgets) (7.34.0)
Requirement already satisfied: jupyterlab-widgets>1.0.0 in /usr/local/lib/python3.11/dist-packages (from ipywidgets) (3.0.13)
Requirement already satisfied: debugpy>1.0 in /usr/local/lib/python3.11/dist-packages (from ipykernel>4.5.1->ipywidgets) (1.8.0)
Requirement already satisfied: jupyter-client>6.1.12 in /usr/local/lib/python3.11/dist-packages (from ipykernel>4.5.1->ipywidgets) (6.1.12)
Requirement already satisfied: matplotlib-inline>0.1 in /usr/local/lib/python3.11/dist-packages (from ipykernel>4.5.1->ipywidgets) (0.1.7)
Requirement already satisfied: nest-asyncio in /usr/local/lib/python3.11/dist-packages (from ipykernel>4.5.1->ipywidgets) (1.6.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from ipykernel>4.5.1->ipywidgets) (24.2)
Requirement already satisfied: psutil in /usr/local/lib/python3.11/dist-packages (from ipykernel>4.5.1->ipywidgets) (5.9.5)
Requirement already satisfied: pyzmq>17 in /usr/local/lib/python3.11/dist-packages (from ipykernel>4.5.1->ipywidgets) (24.0.1)
Requirement already satisfied: tornado>6.1 in /usr/local/lib/python3.11/dist-packages (from ipykernel>4.5.1->ipywidgets) (6.4.2)
Requirement already satisfied: setuptools>18.5 in /usr/local/lib/python3.11/dist-packages (from ipython>4.0.0->ipywidgets) (75.1.0)
Collecting jedi>=0.16 (from ipython>4.0.0->ipywidgets)
  Downloading jedi-0.19.2-py3-none-any.whl.metadata (22 kB)
Requirement already satisfied: decorator in /usr/local/lib/python3.11/dist-packages (from ipython>4.0.0->ipywidgets) (4.4.2)
Requirement already satisfied: pickleshare in /usr/local/lib/python3.11/dist-packages (from ipython>4.0.0->ipywidgets) (0.7.5)
Requirement already satisfied: prompt-toolkit<3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in /usr/local/lib/python3.11/dist-packages (from ipython>4.0.0->ipywidgets) (3.0.50)
Requirement already satisfied: pygments in /usr/local/lib/python3.11/dist-packages (from ipython>4.0.0->ipywidgets) (2.18.0)
Requirement already satisfied: backcall in /usr/local/lib/python3.11/dist-packages (from ipython>4.0.0->ipywidgets) (0.2.0)
Requirement already satisfied: pexpect>4.1 in /usr/local/lib/python3.11/dist-packages (from ipython>4.0.0->ipywidgets) (4.9.0)
Requirement already satisfied: parso>0.4.1 in /usr/local/lib/python3.11/dist-packages (from widgetsnbextension<3.6.0->ipywidgets) (0.8.4)
Requirement already satisfied: parso<0.9.0,>=0.8.4 in /usr/local/lib/python3.11/dist-packages (from jedi>=0.16->ipython>4.0.0->ipywidgets) (0.8.4)
Requirement already satisfied: jupyter-core>4.6.0 in /usr/local/lib/python3.11/dist-packages (from jupyter-client>6.1.12->ipykernel>4.5.1->ipywidgets) (5.7.2)
Requirement already satisfied: python-dateutil>2.1 in /usr/local/lib/python3.11/dist-packages (from jupyter-client>6.1.12->ipykernel>4.5.1->ipywidgets) (2.8.2)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.11/dist-packages (from notebook>4.4.1->widgetsnbextension<3.6.0->ipywidgets) (3.1.6)
Requirement already satisfied: argon2-cffi in /usr/local/lib/python3.11/dist-packages (from notebook>4.4.1->widgetsnbextension<3.6.0->ipywidgets) (23.1.0)
Requirement already satisfied: nbformat in /usr/local/lib/python3.11/dist-packages (from notebook>4.4.1->widgetsnbextension<3.6.0->ipywidgets) (5.10.4)
Requirement already satisfied: nbconvert>5 in /usr/local/lib/python3.11/dist-packages (from notebook>4.4.1->widgetsnbextension<3.6.0->ipywidgets) (7.16.6)
Requirement already satisfied: Send2Trash>1.8.0 in /usr/local/lib/python3.11/dist-packages (from notebook>4.4.1->widgetsnbextension<3.6.0->ipywidgets) (1.8.3)
Requirement already satisfied: terminado>0.8.3 in /usr/local/lib/python3.11/dist-packages (from notebook>4.4.1->widgetsnbextension<3.6.0->ipywidgets) (0.18.1)

```

OUTPUT

Successfully installed jedi-0.19.2

Text:

नकारात्मकता से भरी दुनिया में, दयालुता और करुणा की शक्ति को याद रखना महत्वपूर्ण है। दयालुता के छोटे-छोटे काम किसी के दिन को खुशनुमा बना सकते हैं, उसका मनोबल बढ़ा सकते हैं और सकारात्मकता की लहर पैदा कर सकते हैं जो दूर-दूर तक फैल सकती है। चाहे वह किसी अजनबी को मुस्कुराना हो, ज़रूरत में किसी दोस्त की मदद करना हो या किसी सहकर्मी के लिए कोई विचारशील इशारा हो, दयालुता का हर काम किसी के जीवन में बदलाव लाने की क्षमता रखता है। व्यक्तिगत कार्यों से परे, सकारात्मक बदलाव लाने के लिए सामूहिक प्रयासों में भी अपार शक्ति होती है। जब समुदाय एक-दूसरे का समर्थन करने के लिए एक साथ आते हैं, तो अविश्वसनीय चीजें हो सकती हैं। जमीनी स्तर की पहल से लेकर वैश्विक आंदोलनों तक, लोग सामाजिक और पर्यावरणीय मुद्दों से निपटने के लिए एकजुट हो रहे हैं, सार्थक प्रगति कर रहे हैं और बेहतर भविष्य की उम्मीद जगा रहे हैं। हममें से हर एक के भीतर छिपी ताकत को

Summarize

Summary:

भी छोटे क्यों न हों। अपनी सहज करुणा और सहानुभूति का उपयोग करके, हम दयालुता और सहानुभूति की संस्कृति विकसित कर सकते हैं जो हमारे और हमारे आस-पास के लोगों के जीवन को समृद्ध बनाती है। तो आइए दयालुता की शक्ति को अपनाएँ, और एक समय में एक छोटा सा कार्य करके हैं और सकारात्मकता की लहर पैदा कर सकते हैं जो दूर-दूर तक फैल सकती है। चाहे वह किसी अजनबी को मुस्कुराना हो, ज़रूरत में किसी दोस्त की मदद करना हो या किसी सहकर्मी के लिए कोई विचारशील इशारा हो, दयालुता रहे हैं। हममें से हर एक के भीतर छिपी ताकत को पहचानना भी महत्वपूर्ण है। हम सभी में सकारात्मक प्रभाव डालने की क्षमता है, चाहे हमारे कार्य

5) Conclusion:

Extractive text summarization is a powerful NLP technique that facilitates information retrieval and content compression by selecting the most relevant sentences from a given text. By leveraging statistical, machine learning, and deep learning approaches, extractive summarization has evolved to provide more accurate and context-aware summaries.

With the continuous advancements in AI and NLP, extractive summarization is becoming increasingly effective in various domains, including journalism, legal document analysis, academic research, and business intelligence. Future developments may focus on improving coherence, reducing redundancy, and integrating hybrid approaches to enhance summarization quality further.